MAC0325/5781 COMBINATORIAL OPTIMIZATION: LECTURE NOTES

MARCEL K. DE CARLI SILVA

Contents

Instituto de Matemática e Estatística, Universidade de São Paulo, R. do Matão 1010, 05508-090, São Paulo, SP
E-mail address: mksilva@ime.usp.br.
Date: December 12, 2020 at 09:42pm -0300.

Lecture 1: Optimization Problems

1.1 Basic Terminology of Optimization Problems. An **optimization problem**[i] is an ordered triple $(X, f, \alpha) =: \mathcal{O}$ where[ii]

(i) X is a set, called the **feasible set**[iii] or **feasible region**[iv],
(ii) $f\colon X \to \mathbb{R}$ is a function, called the **objective function**, and
(iii) $\alpha \in \{\pm 1\}$.

For improved readability, it is convenient to abuse notation and write $\mathcal{O}$ as

$$\begin{aligned} &\text{Maximize} && f(x) \\ &\text{subject to} && x \in X \end{aligned} \tag{1.1}$$

if $\alpha = +1$, and we write the easily adapted minimization version of (1.1) if $\alpha = -1$.

Elements of X are the **feasible solutions** or **feasible points** of $\mathcal{O}$; everything else is **infeasible**. The **objective value** of a feasible point $x \in X$ is $f(x)$. We say that the optimization problem $\mathcal{O}$ is **feasible** if $X \neq \varnothing$, and $\mathcal{O}$ is **infeasible**[1] otherwise.

For the next definitions, we will assume that $\alpha = +1$, i.e., $\mathcal{O}$ is a maximization problem. It is straightforward to make the appropriate adaptations to the minimization case where $\alpha = -1$.

The **optimal value** of the optimization problem $\mathcal{O}$ is the extended real number[2]

$$\sup_{x \in X} f(x) \in [-\infty, +\infty] =: \mathrm{OPT}(\mathcal{O}). \tag{1.2}$$

If the optimal value is $+\infty$, then the optimization problem $\mathcal{O}$ is **unbounded**. By convention on the supremum of empty sets, the optimal value is $-\infty$ if and only if $\mathcal{O}$ is infeasible.

A feasible solution x is **optimal** if its objective value $f(x)$ is the optimal value of $\mathcal{O}$, that is, if $f(x) \geq f(y)$ for every $y \in X$. Any such optimal solution is said to **attain** $\sup_{x \in X} f(x)$. If an optimal solution is known to exist[v], we may write 'max' in place of 'sup'. The set of of optimal solutions of $\mathcal{O}$ is denoted $\arg\max_{x \in X} f(x)$.

1.2 Exercise. Work out the adaptations of the definitions for the case $\alpha = -1$ in §1.1. As a bonus, try to do so *without* breaking into the separate cases '$\alpha = +1$' and '$\alpha = -1$'. That is, it is possible to define all terms in a uniform way using the facts that $f(x) \geq f(y) \iff -f(x) \leq -f(y)$ and that[vi]

$$\sup_{x \in X} -f(x) = -\inf_{x \in X} f(x). \tag{1.3}$$

---

[1] Note that we use the terms "feasible" and "infeasible" both for elements/points and for the whole optimization problem.

[2] The set of **extended real numbers** is $\mathbb{R} \cup \{\pm\infty\}$. Recall that the **supremum** of a subset $A \subseteq \mathbb{R}$ of real numbers is the least upper bound for the numbers in A; it is guaranteed to exist, and in fact one may take this as the *defininig property* of real numbers.
If A is **unbounded above**, that is, for each $M \in \mathbb{R}$ there is $a \in A$ such that $a > M$, then set $\sup A := +\infty$; also, we set $\sup \varnothing := -\infty$.
If the set A is defined in the "range-based for" set notation as $A := \{\, \phi(x) : x \in B \}$ for some set B and function $\phi\colon B \to \mathbb{R}$, we adopt the standard syntactic sugar $\sup_{x \in B} \phi(x) := \sup\{\, \phi(x) : x \in B \}$. This syntactic sugar is adopted for other mathematical operators as well.
Analogous observations hold for the **infimum** inf, the greatest lower bound for the elements of some set.

[i] Terms being defined for the first time appear in bold face.

[ii] You must surely have read some mathematical text where you saw some equation of the form $A = B$ and you wondered why such an equation was true, when the author in fact meant to *define* the symbol A as the expression in B. To avoid such frustration, this text will make it *explicit* when the abused equality sign '=' is meant to be a definition (or an assignment operator, as in programming languages), by using $A := B$ when the symbol A is being *defined* as the expression B, and alternatively $B =: A$ when for whatever reason (e.g., flow of text) it is more natural to write the assignment in that form.

[iii] It is **absolutely crucial** that you learn the terminology related to optimization problems and use it appropriately.

[iv] The phrase "region" evokes some geometric imagery, and the use of "feasible region" in place of "feasible set" is used mostly in such cases.

[v] And **only then!**

[vi] If you are scared of 'sup's and 'inf's, do not worry, as they will be used sparingly in this course, and work out the version of (1.3) that has 'max' and 'min'.

1.3 Examples of Optimization Problems. Let us see some examples of optimization problems, using the abuse of notation (1.1):

$$\begin{aligned} &\text{Minimize} && e^x \\ &\text{subject to} && x \in \mathbb{R}, \end{aligned} \tag{1.4}$$

$$\begin{aligned} &\text{Minimize} && x^2 - 2x + 1 \\ &\text{subject to} && x \in \mathbb{R}, \\ & && x^2 \leq 9, \\ & && x \leq 2, \end{aligned} \tag{1.5}$$

$$\begin{aligned} &\text{Minimize} && 1/x \\ &\text{subject to} && x \in \mathbb{R}, \end{aligned} \tag{1.6}$$

$$\begin{aligned} &\text{Maximize} && 1/x \\ &\text{subject to} && x \in (0, 1], \end{aligned} \tag{1.7}$$

$$\begin{aligned} &\text{Maximize} && x \\ &\text{subject to} && x, y \in \mathbb{R}_+, \\ & && \frac{x+y}{2} < \sqrt{xy}, \end{aligned} \tag{1.8}$$

$$\begin{aligned} &\text{Maximize} && 0 \\ &\text{subject to} && n, x, y, z \in \mathbb{N} \\ & && n > 2, \\ & && x^n + y^n = z^n, \\ & && xyz \neq 0. \end{aligned} \tag{1.9}$$

In (1.8), the symbol $\mathbb{R}_+ \coloneqq \{\, \mu \in \mathbb{R} : \mu \geq 0\}$ denotes the set of **nonnegative** real numbers[i]. In (1.9), the symbol $\mathbb{N}$ denotes the set of natural numbers; note that $0 \in \mathbb{N}$.

Some remarks about the examples:

(i) All equations in (1.4) to (1.9) are optimization problems, *except* for (1.6) (why?);
(ii) The feasible region of (1.5) is $[-3, 2]$, which need not be immediately obvious[ii] from the description in (1.5). You should also verify that 1 is the only optimal solution.
(iii) The optimization problem (1.7) has a feasible region that is bounded[1iii] yet its optimal value is $+\infty$, so the optimization problem is *unbounded*.
(iv) The optimization problem (1.4) has an unbounded feasible region, yet its optimal value is 0 (in particular, the optimization problem is *not* unbounded). However, there is no optimal solution (why?).

[1]Recall that a subset $S \subseteq \mathbb{R}$ is **bounded** is there is a number $M \in \mathbb{R}$ such that $S \subseteq [-M, M]$.

[i]Be very careful not to call a nonnegative real number as a **positive** number, which are the elements of $\mathbb{R}_{++} \coloneqq \{\, \mu \in \mathbb{R} : \mu > 0\}$. This distinction will be crucial sometimes in this course. Also, it is hard to justify calling zero a positive number; if you ask me if I have a positive number of private jets, and my currently zero private jets prompts me to answer affirmatively, that would make no sense.

[ii]As we will see extensively in this course, how certain sets are *represented* is paramount to (algorithmic) tractability of problems.

[iii]It is somewhat common to hear students calling bounded sets finite. Please scratch this habit. It is very hard to justify calling the set $[0, 1]$, which has infinitely many elements, finite.

(v) Optimization problems (1.8) and (1.9) are both infeasible. Infeasibility of (1.8) follows from the AM-GM inequality[i]. Infeasibility of (1.9) is equivalent to Fermat's Last Theorem. Since in both the feasible region is $\varnothing$, the optimization problems (1.8) and (1.9) are *the same*![ii]

1.4 Outcomes of an Optimization Problem. Let $(X, f, \alpha) =: \mathcal{O}$ be an optimization problem. Suppose for the sake of definiteness/exposition that $\alpha = +1$. Precisely one of the following holds for the optimal value of $\mathcal{O}$:

(i) the optimal value is $-\infty$; this happens if and only if $\mathcal{O}$ is infeasible;
(ii) the optimal value is $+\infty$; this happens if and only if $\mathcal{O}$ is unbounded;
(iii) the optimal value is finite (i.e., in $\mathbb{R}$); in this case, there might be an optimal solution or not.

Therefore, $\mathcal{O}$ has precisely one of the following four possible *outcomes*:

(1.10)
(i) $\mathcal{O}$ is infeasible;
(ii) $\mathcal{O}$ is unbounded;
(iii) $\mathcal{O}$ has finite optimal value and no optimal solutions[iii];
(iv) $\mathcal{O}$ has an optimal solution.

Solving an optimization problem means determining its outcome in (1.10) and, in the case (1.10)(iv), finding an[1] optimal solution. As a bonus (and we will get this bonus throughout this course), once we determine which outcome applies, one may wish for a *concrete/efficient certificate* that the outcome is correct.

1.5 Theorem (Existence of Optima in the Finite Case). If the feasible region of an optimization problem $\mathcal{O}$ is nonempty and finite[2], then $\mathcal{O}$ has an optimal solution.

Proof. This is essentially a consequence of set-theoretic considerations of the construction of natural numbers; we refer the reader to [5, Section 16]. □

1.6 Representation of the Input. In §1.4, when we refer to solving and optimization problem, we mean to design an algorithm that solves the problem, preferably efficiently. If one casually states the optimization problem as "given an optimization problem (X, f, α), find an optimal solution, if one exists", one inescapably reaches the conclusion that the feasible set X is given as a list of elements (and therefore, since the input is finite, so is the feasible set). In this case, designing a polynomial-time algorithm is trivial. In fact, the following C++11 code provides a linear-time implementation to solve it!

[1] It is very common to hear students mentioning "*the* optimal solution". Even in the case (1.10)(iv) where existence of optimal solution has already been established, proving uniqueness is something else entirely. So, refer to "*an* optimal solution" unless you have already proved uniqueness.
[2] For each $n \in \mathbb{N}$, denote $[n] \coloneqq \{1, \ldots, n\}$; in particular, $[0] = \varnothing$. A set X is **finite** if there is a natural number $n \in \mathbb{N}$ and a bijection $f \colon X \to [n]$.

[i] The inequality for arithmetic and geometric means states that $2\sqrt{xy} \leq x + y$ whenever x and y are nonnegative reals. It is equivalent to the inequality $(\sqrt{x} - \sqrt{y})^2 \geq 0$.
[ii] Both representations of the empty set in (1.8) and (1.9) are quite different. Keep an eye out for these representation results to get a better appreciation of the course, and of computation.
[iii] In this course, this outcome will not occur. You should, however, internalize that this is a possibility, particularly if you take further courses in optimization.

```cpp
#include <iostream>
#include <vector>
#include <algorithm>

struct feas_t {
  std::string name;
  int coolness;
};

int main()
{
  std::vector<feas_t> feas_set { { "Alice",   20 },
                                 { "Bob",     15 },
                                 { "Charlie", 18 }, };
  auto it = std::max_element(feas_set.cbegin(), feas_set.cend(),
                             [](const feas_t &a, const feas_t &b) {
                               return a.coolness < b.coolness; });
  // guaranteed to make <= n comparisons, where n is the size of feas_set
  std::cout << it->name << " is the coolest" << std::endl;
  return 0;
}
```

The difficulty in solving optimization problems is that, in most cases, the feasible set X is *encoded* by some *exponentially smaller representation*. Such representations are necessarily finite, as they are to be the input to algorithms. Even so, they may represent feasible sets that are infinite.

In Combinatorial Optimization, one is mostly interested in optimization problems with a *finite* feasible set[i], especially when the latter is some class of discrete objects encoded by an undirected or directed graph.

Lecture 2: Shortest Paths and Walks

2.1 Digraphs. A **digraph**[ii] is a triple $(V, A, \varphi) =: D$ where

(i) V is a set, called the **vertex**[1] **set** of D,
(ii) A is a set, called the **arc**[2] **set** of D, and
(iii) $\varphi\colon A \to V \times V$ is a function[iii], called the **incidence function** of D.

If $a \in A$ is an arc and $\varphi(a) = (u, v)$ for some $u, v \in V$, then u is the **tail** of a and v is the **head** of a; we also say that a **joins** u to v. Whenever it is clear from context, we may abbreviate the ordered pair (u, v) as uv. If $u = v$, we call a a **loop**; otherwise a is a **non-loop**. A digraph without loops is called **loopless**. If $a, b \in A$ are non-loops such that $\varphi(a) = \varphi(b)$, the arcs a and b are **parallel**.

For a digraph D, we usually denote the vertex set of D by $V(D)$ or V_D, and the arc set of D by $A(D)$ or A_D. Throughout this text,

$$\text{all digraphs are } \textbf{finite}, \tag{2.1}$$

that is, the vertex set and the arc set are both finite.

[1] Therefore, we call each element of V a **vertex** of D.
[2] Therefore, we call each element of A an **arc** of D.

[i] In fact, one can take as a (very vague!) definition of combinatorial optimization as the study of optimization problems with finite feasible sets. However, as we shall soon see, this is not fully accurate.
[ii] The term "directed graph" is also quite common for the same concept. Since we will use it extensively in this text, we adopt the shorter term "digraph" for convenience.
[iii] If X and Y are sets, recall that $X \times Y$, read as "X times Y" or "X Cartesian Y", is the Cartesian product of X and Y, that is, the set of all ordered pairs (x, y) with $x \in X$ and $y \in Y$.

Whenever the incidence function is the identity function (and, consequently, $A \subseteq V \times V$), we may omit φ from the triple[1] $D = (V, A, \varphi)$ and we may just write $D = (V, A)$.

2.2 Walks. A **walk** in a digraph $D \coloneqq (V, A, \varphi)$ is a finite sequence of the form $\langle v_0, a_1, v_1, \ldots, a_\ell, v_\ell \rangle =: W$, where

(i) $\ell \in \mathbb{N}$ is the **length**[i] of W,
(ii) $v_0, \ldots, v_\ell \in V$, and
(iii) $a_1, \ldots, a_\ell \in A$ are such that $\varphi(a_i) = (v_{i-1}, v_i)$ for each $i \in [\ell]$.

We denote $V(W) \coloneqq \{v_0, \ldots, v_\ell\}$ and $A(W) \coloneqq \{a_1, \ldots, a_\ell\}$. We say that W is a walk **from** v_0 **to** v_ℓ, or a (v_0, v_ℓ)**-walk**. By our convention from §2.1, we may also just say v_0v_ℓ-walk.

The walk W is[ii]:

(i) **closed** if $\ell \geq 1$ and $v_0 = v_\ell$,
(ii) a **trail** if all of its arcs are distinct, i.e., the map[iii] $i \in [\ell] \mapsto a_i$ is injective,
(iii) a **path** if all of its vertices are distinct, i.e., the map $i \in \{0, \ldots, \ell\} \mapsto v_i$ is injective,
(iv) a **cycle** if it is a closed trail and the map $i \in [\ell] \mapsto v_i$ is injective.

For $r, s \in V$, an (r, s)**-path** is an (r, s)-walk that is a path.

If $W \coloneqq \langle v_0, a_1, v_1, \ldots, a_\ell, v_\ell \rangle$ is a walk from v_0 to v_ℓ and $Z \coloneqq \langle u_0, b_1, u_1, \ldots, b_k, u_k \rangle$ is a walk from $u_0 = v_\ell$ to u_k, we denote by

$$WZ \coloneqq W \cdot Z \coloneqq \langle v_0, a_1, v_1, \ldots, a_\ell, \underbrace{v_\ell}_{\| \; u_0}, b_1, u_1, \ldots, b_k, u_k \rangle \tag{2.2}$$

the concatenation of W and Z, with the repetition of $v_\ell = u_0$ dropped. So WZ is a walk from v_0 to u_k.

Let $D = (V, A, \varphi)$ be a digraph. If $u, v \in V$ are vertices such that there is a walk from u to v in D, we write $u \rightsquigarrow_D v$, and we say that u **reaches** v (in D), and that v is **reachable from** u (in D). (The index D may be omitted when clear from context.) Note that $\rightsquigarrow_D \subseteq V \times V$ is a binary relation, and the operation in (2.2) shows that $\rightsquigarrow_D$ is transitive[iv].

2.3 Problem (The Shortest Walk Problem). Let $D = (V, A, \varphi)$ be a digraph, and let $c \colon A \to \mathbb{R}$. We refer informally to $c(a)$ as the *cost* of the arc $a \in A$. For a walk $W \coloneqq \langle v_0, a_1, v_1, \ldots, a_\ell, v_\ell \rangle$ in D, denote its (total) cost by

$$c(W) \coloneqq \sum_{i=1}^{\ell} c(a_i). \tag{2.3}$$

The **shortest walk problem** is, **given**[2]

(i) a digraph $D = (V, A, \varphi)$,
(ii) a function[3] $c \colon A \to \mathbb{R}$, and
(iii) vertices $r, s \in V$,

[1] This is probably how you have seen the definition of digraphs before. The reason we keep the incidence function here is that we will need to *name* the arcs, e.g., for the sake of describing algorithms precisely. You may think of these names as the "mathematical version" of "handles" in programming.

[2] It is **crucial** to keep in mind exactly what the input is that is given to an algorithm, since we measure the efficiency of an algorithm by relating its running time to the **size of the input**.

[3] Obviously one cannot represent real numbers exactly on computers, so you may think of c as $c \colon A \to \mathbb{Q}$.

[i] Thus, the length of a walk is the number of *arcs* (not vertices!) traversed by the walk.

[ii] All definitions enumerated are pairwise independent, and they were just clumped together to avoid repetitive text.

[iii] Recall that when we write $f \colon X \to Y$ we mean that f is a function from X to Y. This just specifies the domain and that the image is contained in Y. For the sake of definiteness, let us discuss the function $f \colon \mathbb{R} \to \mathbb{R}$ given by $f(x) \coloneqq x^2$ for each $x \in \mathbb{R}$. This may be written alternatively as $f \colon x \mapsto x^2$. Note that the symbols '$\to$' and '$\mapsto$' are slightly different; whereas '$\to$' describes domain and codomain, $\mapsto$ defines the function for each element in the domain (think lambdas in Python or C++1x). In this case, the function has a name, f, and its domain and codomain were already specified. If we are dealing instead with an anonymous function, we might just write it as $x \in \mathbb{R} \mapsto x^2$; note how we explicitly mention the domain. This is the same function as before.

[iv] Recall that a binary relation $\sim$ on a set V is a subset of the Cartesian product $V \times V$. In this case, it is conventional to write $(u, v) \in \sim$ as $u \sim v$. The binary relation $\sim$ is **transitive** if, for every $u, v, w \in V$, the conditions $u \sim v$ and $v \sim w$ imply $u \sim w$.

FIGURE 2.1. There are 2^{n-1} paths from r to s.

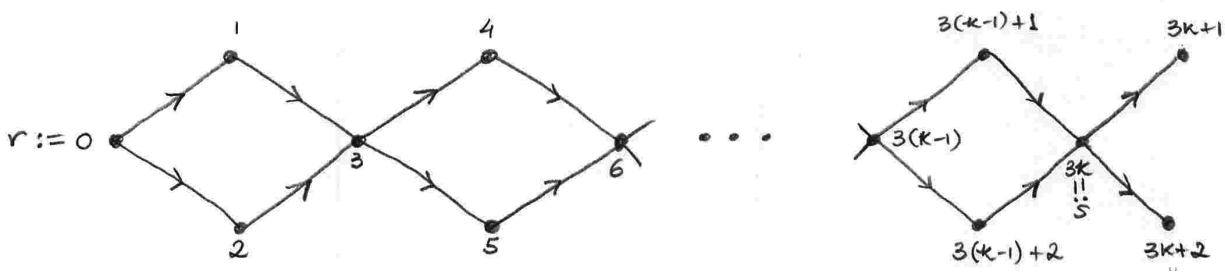


FIGURE 2.2. There are 2^k paths from r to s. Note that $2^k = 2^{n/3-1} = \frac{1}{2}(\sqrt[3]{2})^n \approx \frac{1}{2}(1.26)^n$.

solve[i] the optimization problem

(SWP)
$$\begin{array}{ll} \text{Minimize} & c(W) \\ \text{subject to} & W \text{ is an } (r,s)\text{-walk in } D. \end{array}$$

2.4 Problem (The Shortest Path Problem). A problem related to the shortest walk problem from Problem 2.3 and probably more well-known is the **shortest path problem**, described as follows. **Given**

(i) a digraph $D = (V, A, \varphi)$,
(ii) a function $c\colon A \to \mathbb{R}$, and
(iii) vertices $r, s \in V$,

solve the optimization problem

(SPP)
$$\begin{array}{ll} \text{Minimize} & c(P) \\ \text{subject to} & P \text{ is an } (r,s)\text{-path in } D. \end{array}$$

Note that

(2.4) either (SPP) is infeasible or it has an optimal solution.

Indeed, the feasible set is finite[ii], so (SPP) has an optimal solution by Theorem 1.5 whenever it is feasible.

2.5 Algorithmic Efficiency and Representation. Throughout the course, we will deal with optimization problems defined over graphs and digraphs. It is convenient to refer to the number of vertices as n and the number of arcs as m, whenever the digraph this refers to is obvious. In order for an algorithm on digraphs or graphs to be **efficient**, we require its running time to be a polynomial in the size of the input. For the sake of encoding a digraph for input into an algorithm, one may take V to be $[n]$, and so the size of the input (by considering an adjacency list representation for D) is $\Theta(n+m) + \text{size}(c)$, where size$(c)$ is "essentially"[iii] $\sum_{a\in A} \lg|c_a|$.

Thus, an algorithm that solves the shortest path/walk problem must run in time polynomial in n, m, and size(c). However, the feasible set for (SPP), while finite, may have *exponential* size in n; see the examples in Figures 2.1 and 2.2. Thus, a simple-minded approach such as the one in §1.6 does not work. This illustrates the paramount dependence on the form of the *representation* of the input to an algorithm.

In this course, our interest is in devising polynomial-time algorithms, not necessarily very fast ones. Thus, we will *not* focus on designing extremely efficient (and intricate) data structures.

[i] In the sense of §1.4.
[ii] Verify this. For instance, prove that every path has length $\leq n-1$, and that there are at most $(nm)^{k+1}$ walks of length k.
[iii] We say "essentially" because one needs to consider slightly annoying (and irrelevant) details, such as a sign bit, ceilings, and so on.

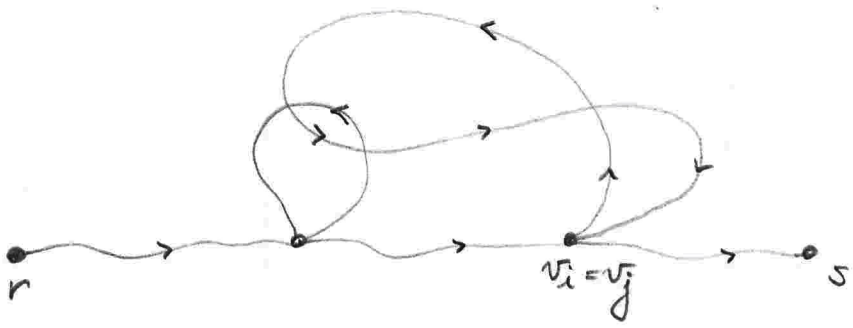


FIGURE 2.3. Snipping the closed sub-walk from v_i to v_j.

2.6 The Shortest Path Problem is NP-hard. Problem (SPP) is more well known and more desirable to solve. However, this problem is NP-hard[i]. Let us discuss this briefly.

Recall that a path P in a digraph $D = (V, A, \varphi)$ is **Hamiltonian** if $V(P) = V$, and recall that the problem of deciding whether an input digraph has a Hamiltonian path between two prescribed vertices is NP-complete[ii]. Next note that, for a digraph $D = (V, A, \varphi)$ and distinct vertices r, s, there is a Hamiltonian path from r to s in D if and only if the optimal value of (SPP), when we set[1][iii] $c := -\mathbb{1}$, is $-(n-1)$. Thus, if there is a polynomial-time algorithm for (SPP), then there is a polynomial-time algorithm for the NP-complete problem of deciding Hamiltonicity (with prescribed ends) of a given digraph, so the shortest path problem is NP-hard.

2.7 We will see that, under "natural" assumptions, the shortest path problem and the shortest walk problem are equivalent, and then we will show how to solve the latter. We will also define precisely the notion of equivalence between optimization problems. In order to do this, we will have to study the "pathological" outcomes of the shortest walk problem, i.e., outcomes without an optimal solution.

2.8 Negative Cycles and Unboundedness. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. A cycle $C = \langle v_0, \ldots, v_\ell \rangle$ in D is a **negative cycle** (with respect to c)[iv] if $c(C) < 0$. Suppose that $r \rightsquigarrow v_0$ and $v_\ell \rightsquigarrow s$. Let R be a walk from r to v_0, and let S be a walk from v_ℓ to s. For each $k \in \mathbb{N}$, the walk

$$W_k := R \cdot \Big(\prod_{i \in [k]} C \Big) \cdot S \tag{2.5}$$

is feasible for (SWP) (see Figure 2.3) and its objective value is $c(W_k) = c(R) + kc(C) + c(S)$. Since $c(C) < 0$ and k can be taken arbitrarily high, the optimal value of (SWP) is $-\infty$, i.e., the optimization problem is unbounded.

2.9 Proposition. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. Suppose there is a negative cycle C and that some $v \in V(C)$ satisfies $r \rightsquigarrow v \rightsquigarrow s$. Then (SWP) is unbounded.

Proof. Immediate from §2.8. □

[1] Denote the vector of all-ones by $\mathbb{1}$, where the domain is implicit in the context.

[i] You have all taken basic complexity in your Analysis of Algorithms course, and we will make use of your previous knowledge.
[ii] It is possible that you have seen a slight variation of this, possibly on (undirected) graphs and without prescribing the vertices linked by the Hamiltonian path. It is a simple exercise to verify that the version stated here is also NP-complete.
[iii] This is somewhat equivalent to formulating the problem of finding a maximum-cost rs-path, where each arc has unit cost; see (1.3).
[iv] When definitions use terms in parentheses, these are usually clear from context whenever the definition is applied, and so the part in parentheses omitted whenever this is the case.

Lecture 3: Shortest Walks and Negative Cycles

3.1 Theorem ("Walk Yields Path or Negative Cycle"). Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. Let W be an rs-walk in D. Then at least one of the following holds:

(3.1) (i) there is an rs-path P in D with cost $c(P) \leq c(W)$ and $V(P) \subseteq V(W)$;
(ii) there is a negative cycle C in D such that $V(C) \subseteq V(W)$.

Proof. Let k be the length of W. Then the optimization problem

$$\begin{aligned} &\text{Minimize} && \text{length of } W' && \text{(3.2a)}\\ &\text{subject to} && W' \text{ is an } rs\text{-walk in } D \text{ of length} \leq k, && \text{(3.2b)}\\ & && c(W') \leq c(W), && \text{(3.2c)}\\ & && V(W') \subseteq V(W), && \text{(3.2d)} \end{aligned}$$

is feasible. Since the feasible set is nonempty and finite (why?), Theorem 1.5 shows that there is an optimal solution $W^* := \langle v_0, a_1, v_1, \ldots, a_\ell, v_\ell \rangle$ for (3.2). If W^* is a path, (3.2c) and (3.2d) show that (3.1)(i) holds. So suppose that W^* is not a path. Then

$$I := \{ (i,j) : i, j \in \{0, \ldots, \ell\},\ i < j,\ v_i = v_j \} \neq \varnothing. \tag{3.3}$$

Denote

$$W^*_{ij} := \langle v_i, a_{i+1}, v_{i+1}, \ldots, a_j, v_j \rangle \qquad \text{for each } ij \in I.$$

Suppose for the sake of contradiction that

$$c(W^*_{ij}) \geq 0 \tag{3.4}$$

for some $ij \in I$. Then the rs-walk (see Figure 2.3)

$$Z := \langle \underbrace{v_0}_{\| r}, a_1, v_1, \ldots, \underbrace{v_i}_{\| v_j}, a_{j+1}, v_{j+1}, \ldots, a_\ell, \underbrace{v_\ell}_{\| s} \rangle$$

has

$$c(Z) = c(W^*) - c(W^*_{ij}) \overset{(3.4)}{\leq} c(W^*) \overset{(3.2c)}{\leq} c(W),$$

and

$$V(Z) \subseteq V(W^*) \overset{(3.2d)}{\subseteq} V(W),$$

i.e., Z is feasible for (3.2), yet Z has strictly smaller length than W^*, a contradiction. Hence,

$$c(W^*_{ij}) < 0 \qquad \text{for each } ij \in I. \tag{3.5}$$

Let (i^*, j^*) be an optimal solution to

$$\begin{aligned} &\text{Minimize} && |i - j| \\ &\text{subject to} && (i,j) \in I, \end{aligned} \tag{3.6}$$

which exists by Theorem 1.5. Then

$$W^*_{i^* j^*} \text{ is a cycle.} \tag{3.7}$$

Indeed, if $v_i = v_j$ for some indices i, j such that $i^* < i < j \leq j^*$, then the objective value of $(i,j) \in I$ in (3.6) is smaller than that of (i^*, j^*), a contradiction. Thus, by (3.5), (3.7), and (3.2d), it follows that (3.1)(ii) holds[1]. □

3.2 Corollary (Equivalence of Reachability by Walks and Paths). Let $D = (V, A, \varphi)$ be a digraph, and let $r, s \in V$. Then there is an rs-path in D if and only if there is an rs-walk in D.

[1] Once you understand this proof, you most likely will feel an uncontrollable urge to dismiss it and say: if your current rs-walk is not a path, find a closed walk within it, cancel/remove it, and **repeat**, until you end up with a path or a negative cycle. This argument is absolutely fine, and it can be made rigorous, and it matches our computational way of thinking. **However**, we will see later how slight variations of this argument (especially the "repeat" part) may fail. This is why I am writing this proof in this rather unintuitive order, so that you can start to get used to it.

Proof. Necessity:[i] Suppose there is an rs-path W in D. Since every path is a walk, W is an rs-walk in D.

Sufficiency: Suppose there is an rs-walk in D. Set $c := 0 \in \mathbb{R}^A$[1]. By Theorem 3.1, one of the options[ii] in (3.1) holds. However, (3.1)(ii) is impossible, since every walk has cost 0. Hence, (3.1)(i) shows that there is an rs-path in D. □

3.3 Corollary. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. Then (SWP) is feasible if and only if (SPP) is feasible.

Proof. Immediate from Corollary 3.2. □

3.4 Corollary ("Unboundedness $\iff$ Negative Cycle"). Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. Then the shortest walk problem (SWP) is unbounded if and only if there is a negative cycle C in D and a vertex $v \in V(C)$ such that $r \rightsquigarrow v \rightsquigarrow s$.

Proof. Sufficiency: This is Proposition 2.9.

Necessity: Suppose that (SWP) is unbounded. In particular, (SWP) is feasible, and so is the shortest path problem (SPP) by Corollary 3.3. By (2.4),

(3.8) the shortest path problem (SPP) has an optimal solution.

Let

$$\nu^* := \min\{\, c(P) : P \text{ an } rs\text{-path}\} \overset{(3.8)}{\in} \mathbb{R} \tag{3.9}$$

be the optimal value of (SPP). Since the shortest walk problem (SWP) is unbounded, there is a feasible solution W for (SWP) with objective value

$$c(W) < \nu^* \overset{(3.9)}{\in} \mathbb{R}. \tag{3.10}$$

By Theorem 3.1, one of the options in (3.1) holds. If (3.1)(i) holds, there is an rs-path P' with cost

$$c(P') \leq c(W) \overset{(3.10)}{<} \nu^* \overset{(3.9)}{=} \min\{\, c(P) : P \text{ an } rs\text{-path}\},$$

a contradiction. Hence, (3.1)(ii) holds, i.e., there is a negative cycle C with $V(C) \subseteq V(W)$. Let $v \in V(C)$. Then $v \in V(W)$ whence $r \rightsquigarrow v \rightsquigarrow s$. □

3.5 Certificates of Unboundedness. Corollary 3.4 shows that, whenever the shortest walk problem (SWP) is unbounded, there is a short[2] and obvious[3] *certificate* of this fact, namely, the negative cycle together with walks from r and to s. Note that such certificate exists if and only if the problem is unbounded. Also, this certificate is an *object* that an algorithm can return to its caller/client. In particular, note that it is not the case that the algorithm will return a string that says "Look, I checked that there are rs-walks of objective value arbitrarily close to $-\infty$, trust me." Rather, by returning this object/certificate, the client can verify for itself that the problem is unbounded, *without repeating the procedure that was used to find such a certificate*![iii]

3.6 Corollary. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. If the shortest walk problem (SWP) is feasible and not unbounded, then for every rs-walk W in D there is an rs-path in D such that $c(P) \leq c(W)$.

---

[1] Recall that, if A and B are sets, then B^A denotes the set of all functions from A to B.

[2] This can be made mathematically precise, in that the certificate can be encoded in size polynomial in the size of the input.

[3] This is not mathematically precise, but your boss can understand that a negative cycle certifies that the problem is unbounded.

[i] When we have a statement of the form "X if and only if Y", it is conventional (though not required) to treat X as a "desired" statement, and Y as some equivalent, though not obviously desirable, condition. Thus, "necessity" refers to the necessity of condition Y, i.e., if the desired statement X holds, then Y necessarily holds as well. This is also known as the 'only if' part of "X if and only if Y", so that "X only if Y" means that "X implies Y". "Sufficiency" refers to the sufficiency of condition Y for X, i.e., if the sufficient condition Y holds, then the desired statement X holds. This is also known as the 'if' part of "X if and only if Y", so that "X if Y" means that "Y implies X".

[ii] Note that we are using Theorem 3.1, which involves arc costs, to derive a purely graph-theoretical (and optimization-free) result. Clearly, this result could be derived without resorting to Theorem 3.1. However, we already have Theorem 3.1 in our API, and it is more elegant to rely on that as a black-box than to poke around the *proof*/implementation of Theorem 3.1. You should think of this as the mathematical version of code encapsulation.

[iii] That is why it is important to note that the certificate is a concrete object that can be returned by an algorithm.

Proof. Let W be an rs-walk in D. By Theorem 3.1, one of the options in (3.1) holds. However, (3.1)(ii) is impossible by Proposition 2.9. So there is an rs-path P in D with objective value $c(P) \leq c(W)$. □

3.7 Corollary. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. Then the shortest walk problem (SWP) is either infeasible, or unbounded, or it has an optimal solution. In the latter case, there is an optimal solution which is an rs-path.

Proof. Suppose

(3.11) (SWP) is feasible and not unbounded.

Since every rs-path is feasible in (SWP), Corollary 3.6 shows that, in the shortest walk problem, it suffices to look at the feasible solutions that are paths. In other words, the shortest walk problem (SWP) is equivalent[1] to the shortest path problem (SPP). However, (SPP) is feasible by (3.11) and Corollary 3.3, and so it has an optimal solution by (2.4). Hence, the equivalent problem (SWP) also has an optimal solution, and any optimal solution for (SPP) is optimal for (SWP) as well. □

3.8 Corollary 3.7 shows that the "pathological" case (1.10)(iii) cannot happen for the shortest walk problem (SWP). Also, Corollary 3.3 shows that infeasibility occurs simultaneously for shortest paths and shortest walks. Finally, Corollaries 3.7 and 3.4 show that, in the absence of negative cycles reachable from r and that reach s, shortest paths and shortest walks are equivalent.

3.9 Reachability and Basic Graph Traversal. Let $D = (V, A, \varphi)$ be a digraph and let $r \in V$. The set $R \coloneqq \{\, v \in V : r \rightsquigarrow v \,\}$ of all vertices reachable from r can be computed in $O(n + m)$ time by basic graph traversal algorithms, called breadth-first search (BFS) and depth-first search (DFS). Not only that, these algorithms return extra information (in the form of a vector of predecessors) that enables recovery of paths from r to each vertex in R in $O(n + m)$ time. We will *not* cover these algorithms in this course, and refer the reader to the exposition from [3, Ch. 22]. (Also, such an algorithm may be derived from the algorithm we will describe later for the shortest walk problem, albeit with a worse running time.)

Similarly, for $s \in V$, the set $S \coloneqq \{\, v \in V : v \rightsquigarrow s \}$ of vertices that reach s can be computed in $O(n + m)$ time, as well as paths that certify membership in S. This can be seen by modifying the aforementioned algorithms, or by applying them as black boxes on the **reverse** of D defined as

$$D^{-1} \coloneqq (V, A, \psi), \tag{3.12}$$

where for each $a \in A$ we take $\psi(a) \coloneqq (\varphi(a))^{-1}$; for an ordered pair (u, v), we denote

$$(u, v)^{-1} \coloneqq (v, u). \tag{3.13}$$

3.10 Cuts. Let $D = (V, A, \varphi)$ be a digraph, and let $R \subseteq V$. Denote $\overline{R} \coloneqq V \setminus R$. Define

$$\delta^{\text{out}}_D(R) \coloneqq \big\{\, a \in A : \varphi(a) \in R \times \overline{R} \big\}, \tag{3.14}$$

the set of arcs that **leave** R. (As usual, the index D may be omitted when clear from the context.) Define

$$\delta^{\text{in}}_D(R) \coloneqq \delta^{\text{out}}_D(\overline{R}), \tag{3.15}$$

the set of arcs that **enter** R. See Figure 3.1.

Sets of these forms are referred *informally* as cuts, and the arguments R to δ^{out} and δ^{in} in (3.14) and (3.15) are sometimes addressed as shores.

Let $r, s \in V$. If $r \in R$ and $s \notin R$, then we call the set $\delta^{\text{out}}(R)$ an (r, s)**-cut**. That is, the set of rs-cuts is $\{\, \delta^{\text{out}}(R) : r \in R \subseteq V \setminus \{s\} \}$.

The next exercise should be intuitively obvious.

3.11 Exercise (Every rs-walk meets every rs-cut). Let $D = (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let W be an rs-walk, and let $R \subseteq V$ such that $r \in R$ and $s \notin R$. Prove[i] that $A(W) \cap \delta^{\text{out}}(R) \neq \varnothing$.

3.12 Exercise. Let $D = (V, A, \varphi)$ be a digraph and let $r \in V$. Set $R \coloneqq \{\, v \in V : r \rightsquigarrow v \}$. Prove that $\delta^{\text{out}}(R) = \varnothing$.

3.13 Corollary. Let $D = (V, A, \varphi)$ be a digraph, and let $r, s \in V$. Then there is an rs-walk in D if and only if there is no subset $R \subseteq V$ such that $r \in R$, $s \notin R$, and $\delta^{\text{out}}(R) = \varnothing$.

[1] This may be a bit vague at this point, but soon we will make this argument more general and more rigorous.

[i] *Hint:* Let $W =: \langle v_0, a_1, v_1, \ldots, a_\ell, v_\ell \rangle$ and let $k \in \{0, \ldots, \ell\}$ be maximum such that $v_k \in R$.

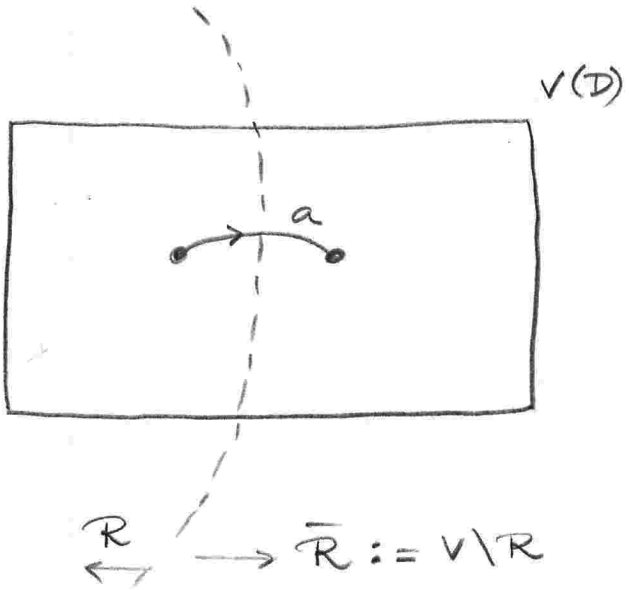


FIGURE 3.1. Illustration for an arc in $\delta^{\text{out}}(R)$.

Proof. Necessity: Suppose there is an rs-walk W in D. Let $R \subseteq V$ such that $r \in R$ and $s \notin R$. Then

$$\begin{aligned} \varnothing \neq A(W) \cap \delta^{\text{out}}(R) \quad & \text{by Exercise 3.11} \\ \subseteq \delta^{\text{out}}(R), & \end{aligned}$$

whence $\delta^{\text{out}}(R) \neq \varnothing$.

Sufficiency: We will prove the contrapositive[1]. Suppose that there is no rs-walk in D. Define $R \coloneqq \{ v \in V : r \rightsquigarrow v \}$. Then $r \in R$ and $s \notin R$. Moreover, $\delta^{\text{out}}(R) = \varnothing$ by Exercise 3.12. □

3.14 Infeasibility and Certificates. Let $D = (V, A, \varphi)$ be a digraph, and let $r, s \in V$. Let $R \coloneqq \{ v \in V : r \rightsquigarrow v \}$ be the set of all vertices reachable from r, computed in $O(n + m)$ time as in §3.9. If $s \in R$, then (SWP) and (SPP) are feasible, and a feasible solution can be recovered in linear time. If not, then R proves infeasibility for (SWP) and (SPP) by way of Corollary 3.13.

If your boss asks you to find an rs-walk and no such walk exists, the set R returned above makes it **obvious** that there is no (r, s)-walk in D. As in §3.5, the certificate is an *object* (namely, the set of vertices R) that can be returned by the algorithm to the caller. To be fair, verification by the caller that this is a certificate takes linear time too (the caller must verify that no arc leaves R), however the caller does not need to traverse the digraph for the verification.

3.15 Certificates and Existential Quantifiers. It is easy to get confused and to think that every "if and only if" result provides a certificate of a fact. Consider the following equivalences for a digraph $D = (V, A, \varphi)$ and vertices $r, s \in V$, arising from Corollary 3.2 and Corollary 3.13, respectively:

$$\text{there is no } rs\text{-walk in } D \iff \text{there is no } rs\text{-path in } D, \tag{3.16}$$

$$\iff \text{there is } R \subseteq V \text{ with } r \in R,\ s \notin R, \text{ and } \delta^{\text{out}}(R) = \varnothing. \tag{3.17}$$

The top equivalence (3.16) does *not* provide a certificate: to certify that no rs-walk exists, it says "look, I conducted a thorough, exhaustive search and yet I could not find an rs-path, so no rs-walk exists, trust me." The bottom equivalence (3.17) provides a certificate: **it proves that something does NOT exist by exhibiting something that exists (!!!)**, and therefore can be exhibited.

[1]Recall that the contrapositive of "X implies Y" is "not Y implies not X"; proving one of these also proves the other.

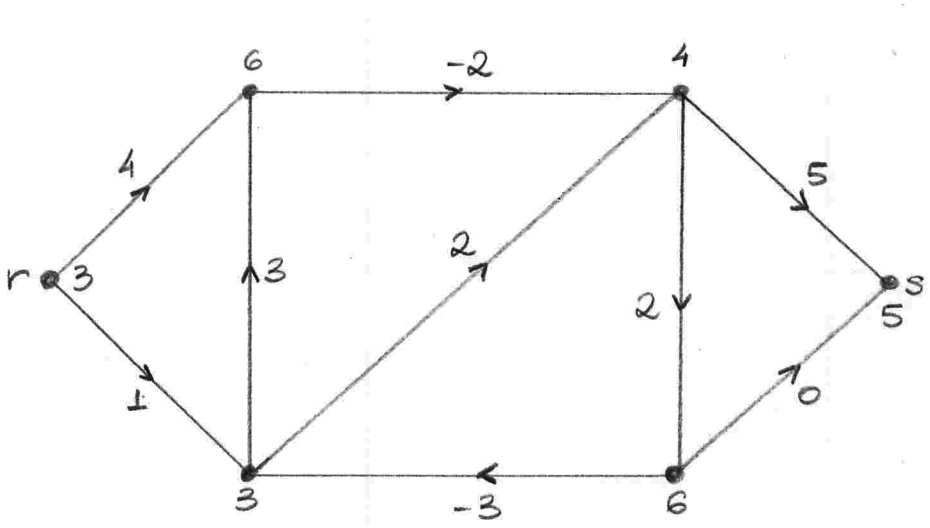


FIGURE 4.1. A feasible potential for the given arc costs.

LECTURE 4: SHORTEST WALKS AND FEASIBLE POTENTIALS

4.1 Feasible Potentials. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. We say that a function $y\colon V \to \mathbb{R}$ is a **feasible potential** (with respect to c) if[1]

$$y(v) \leq y(u) + c(a) \qquad \text{for each } a \in A, \text{ where } uv \coloneqq \varphi(a). \tag{4.1}$$

See an example at Figure 4.1.

Note the simple fact that, for a function $y\colon V \to \mathbb{R}$ and $\mu \in \mathbb{R}$, we have

$$y \text{ is a feasible potential} \iff y + \mu\mathbb{1} \text{ is a feasible potential.} \tag{4.2}$$

For this reason, when considering whether a feasible potential exists, one may choose a vertex $r \in V$ and a prescribed $\alpha \in \mathbb{R}$, and consider the (apparently) more restricted question of whether there is a feasible potential y with $y(r) = \alpha$.

4.2 Theorem. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. Let $y\colon V \to \mathbb{R}$ be a feasible potential. If W is an rs-walk in D, then

$$c(W) \geq y(s) - y(r). \tag{4.3}$$

Proof. Set $W =: \langle v_0, a_1, v_1, \ldots, a_\ell, v_\ell \rangle$. By using that y is a feasible potential and (4.1), we find that

$$c(a_i) \geq y(v_i) - y(v_{i-1}) = -y(v_{i-1}) + y(v_i) \qquad \text{for each } i \in [\ell], \tag{4.4}$$

since $\varphi(a_i) = (v_{i-1}, v_i)$ for each $i \in [\ell]$. Hence,

$$\begin{aligned} c(W) &= \sum_{i=1}^{\ell} c(a_i) \geq \sum_{i=1}^{\ell} \Big(-y(v_{i-1}) + y(v_i)\Big) && \text{by (4.4)} \\ &= \big(-y(v_0) + y(v_1)\big) + \big(-y(v_1) + y(v_2)\big) + \cdots + \big(-y(v_{\ell-1}) + y(v_\ell)\big) \\ &= -y(v_0) + y(v_\ell) && \text{by telescoping} \\ &= y(s) - y(r). \end{aligned} \tag{4.5}$$

This proves (4.3). □

[1] We will see that such feasible potentials are extremely useful in the study of the shortest walk problem. Yet it is not clear at all what motivates one to define these beasts. For now, we will disregard the question of how one could have come up with the definition, and later we will see that one could have arrived at the definition in an almost "mechanical" way. Hint: this has to do with linear optimization duality.

4.3 Corollary. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. Let W^* be an rs-walk in D, and suppose that $y\colon V \to \mathbb{R}$ is a feasible potential such that

$$c(W^*) = y(s) - y(r). \tag{4.6}$$

Then W^* is an optimal solution for (SWP).

Proof. Let W be an rs-walk in D. Then

$$\begin{aligned} c(W) &\geq y(s) - y(r) && \text{by Theorem 4.2} \\ &= c(W^*) && \text{by (4.6).} \end{aligned}$$

Hence, W^* is optimal for (SWP). □

4.4 Upper Bounds, Lower Bounds, and Certificates. Let us focus on the shortest walk problem (SWP). This is a minimization problem, and it is *easy*[i] to obtain upper bounds for its optimal value ν^*: it suffices to find *any* rs-walk, that is, if W is an rs-walk, then $\nu^* \leq c(W)$. Obtaining lower bounds for ν^* *seems much harder*: *a priori*, to test if $L \in \mathbb{R}$ is a lower bound for ν^*, i.e., if $L \leq \nu^*$, it seems that we need to **check that every feasible solution** has objective value $\geq L$. (Naturally, for a maximization problem, the roles of upper and lower bounds in this discussion are reversed.)

Theorem 4.2 shows that it is possible to obtain lower bounds for ν^* **without having to check every feasible solution** for (SWP). In fact, in the case of Corollary 4.3, one can use the feasible potential y (the fact that y is a feasible potential can be checked in linear time) as a **certificate of optimality** of a feasible solution.

This. Is. Awesome.

More awesome yet is that we will be able to certify optimality for the shortest walk problem by using feasible potentials, in the format of Corollary 4.3, *whenever an optimal solution exists*.

4.5 Corollary. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function. If $y\colon V \to \mathbb{R}$ is a feasible potential, then there is no negative cycle in D.

Proof. Let $C = \langle v_0, \ldots, a_\ell, v_\ell \rangle$ be a closed walk in D. Then

$$\begin{aligned} c(C) &\geq y(v_\ell) - y(v_0) && \text{by Theorem 4.2} \\ &= 0 && \text{since } v_\ell = v_0. \end{aligned}$$

Hence, $c(C) \geq 0$. □

4.6 Restricting the Reachable Set. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. Suppose there is a shortest rs-walk, i.e., (SWP) has an optimal solution. We mentioned in §4.4 that we will provide a feasible potential as certificate of optimality of an optimal rs-walk found. By Proposition 2.9, there is no negative cycle reachable from r and that reaches s. However, there might be a negative cycle reachable from r (and which does not reach s), or there might be a negative cycle that reaches s (and which is not reachable from r); see Figure 4.2. In either case, Corollary 4.5 shows it is impossible to find a feasible potential for the whole digraph. If we are to have any chance of finding such a feasible potential to certify optimality, we will have to restrict ourselves to the set of vertices reachable from r and that reach s, and then focus on the corresponding *subdigraph*.

4.7 Subdigraphs. A digraph $D' \coloneqq (V', A', \varphi')$ is a **subdigraph** of a digraph $D \coloneqq (V, A, \varphi)$ if $V' \subseteq V$, $A' \subseteq A$, and $\varphi' = \varphi\restriction_{A'}$, i.e., φ' is the **restriction**[ii] of φ to A'; that is, $\varphi'(a) = \varphi(a)$ for each $a \in A'$.

In this case, we say that D is a **superdigraph** of D'.

Let $U \subseteq V$. We say that an arc $a \in A$ is **induced** by U if $\varphi(a) \in U \times U$, i.e., if both its tail and its head lie in U. The set of all arcs in A induced by U is denoted by $A_\varphi[U]$ or $A[U]$. The **subdigraph** of D **induced by** U is $D[U] \coloneqq (U, A_\varphi[U], \varphi\restriction_{A_\varphi[U]})$.

[i] This is not to say that it is always easy to find a feasible solution for an optimization problem described implicitly; refer back to our discussion about Fermat's Last Theorem in §1.3.

[ii] Recall that a function $f\colon X \to Y$ is a relation $f \subseteq X \times Y$, i.e., a subset of the Cartesian product $X \times Y$, with the property that, for each $x \in X$, there is precisely one element $y \in Y$ such that $(x, y) \in f$, and this element y is denoted by $f(x)$. Thus, the restriction of f to $S \subseteq X$ is precisely $f \cap (S \times B)$.

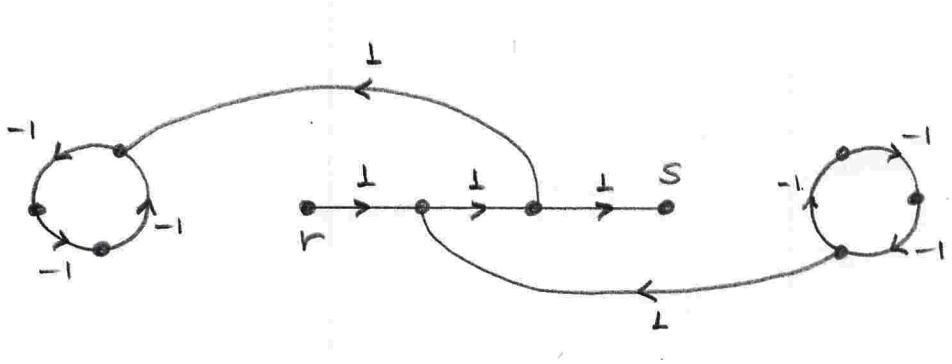


FIGURE 4.2. A digraph with a negative cycle reachable from r (and which does not reach s) and another negative cycle that reaches s (and which is not reachable from r).

4.8 Exercise. Let $D = (V, A, \varphi)$ be a digraph and let $r, s \in V$. Let $R \subseteq V$ such that $r \in R$ and $\delta_D^{\text{out}}(R) = \varnothing$. Let $S \subseteq V$ such that $s \in S$ and $\delta_D^{\text{in}}(S) = \varnothing$. Prove[i] that, if W is an rs-walk in D, then W is an rs-walk in $D[R \cap S]$.

4.9 Corollary. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. Let $R \subseteq V$ such that $r \in R$ and $\delta_D^{\text{out}}(R) = \varnothing$. Let $S \subseteq V$ such that $s \in S$ and $\delta_D^{\text{in}}(S) = \varnothing$. Let W^* be an rs-walk in D. If $y\colon R \cap S \to \mathbb{R}$ is a feasible potential for $D[R \cap S]$ such that $c(W^*) = y(s) - y(r)$, then W^* is an optimal solution for (SWP).

Proof. By Exercise 4.8, the optimization problems (SWP) over D and over $D[R \cap S]$ are *equal*[ii]. The result now follows from Corollary 4.3, which proves optimality of W^* in the shortest walk problem over $D[R \cap S]$. □

4.10 The Shortest Walk Problem with Bounded Length. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. In order to devise an algorithm for the shortest walk problem (SWP), we use an incremental approach by solving the same problem with an extra constraint that bounds the length of the feasible walks. Define, for each $v \in V$ and $t \in \mathbb{N}$, the optimization problem

$$(\text{SWP}_{\leq t}(D, r, v)) \qquad \begin{array}{ll} \text{Minimize} & c(W) \\ \text{subject to} & W \text{ is an } rv\text{-walk in } D \text{ of length} \leq t. \end{array}$$

When $t = 0$, this optimization problem is easy to solve. In order to get optimal solutions for $t > 0$ from optimal solutions to $t - 1$, we rely on the following optimal substructure result to provide a dynamic programming approach.

4.11 Lemma (Sub-walks of Shortest Walks Are Shortest, a.k.a. "Optimal Substructure"). Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, v \in V$. Let $t \in \mathbb{N}$. If $W^* \coloneqq \langle v_0, \ldots, a_\ell, v_\ell \rangle$ is an optimal solution for $(\text{SWP}_{\leq t}(D, r, v))$, then

$$W_{0i}^* \coloneqq \langle v_0, \ldots, a_i, v_i \rangle \text{ is an optimal solution for } (\text{SWP}_{\leq i}(D, r, v_i)), \quad \text{for each } i \in \{0, \ldots, \ell\}. \tag{4.7}$$

Proof. Let $i \in \{0, \ldots, \ell\}$. Define $W_{i\ell}^* \coloneqq \langle v_i, \ldots, a_\ell, v_\ell \rangle$, so that $W^* = W_{0i}^* \cdot W_{i\ell}^*$ whence

$$c(W^*) = c(W_{0i}^*) + c(W_{i\ell}^*). \tag{4.8}$$

Clearly W_{0i}^* is feasible for $(\text{SWP}_{\leq i}(D, r, v_i))$, so if W_{0i}^* is not optimal for $(\text{SWP}_{\leq i}(D, r, v_i))$, there is a walk W' from r to v_i of length $\leq i$ such that

$$c(W') < c(W_{0i}^*). \tag{4.9}$$

But then $W' \cdot W_{i\ell}^*$ is a walk from r to $v_\ell = v$ with length $\leq i + (\ell - i) = \ell \leq t$ and cost

$$c(W' \cdot W_{i\ell}^*) = c(W') + c(W_{i\ell}^*)$$

[i] Let $W = \langle v_0, a_1, v_1, \ldots, a_\ell, v_\ell \rangle$ be an rs-walk in D. It suffices to prove that $V(W) \subseteq R \cap S$. To prove $V(W) \subseteq R$, use Corollary 3.13 and $\langle v_0, a_1, v_1, \ldots, a_i, v_i \rangle$. A symmetric argument shows that $V(W) \subseteq S$.

[ii] Not just equivalent, but identical.

$$\begin{aligned} &= c(W') + c(W^*) - c(W^*_{0i}) && \text{by (4.8)} \\ &< c(W^*) && \text{by (4.9)}, \end{aligned}$$

a contradiction since W^* is optimal for (SWP$_{\le t}(D, r, v)$). □

4.12 The Bellman-Ford Algorithm. We describe[1] in Algorithm 4.1 a dynamic programming algorithm that solves the shortest walk problem from Problem 2.3, based on §4.10. You should peruse the algorithm before proceeding, and verify that the algorithm runs in polynomial time.

Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$.

The algorithm performs n' iterations for the **for** loop in Line 14, where n' is the size of the set V' of vertices reachable from r and that reach s. For each index t in a subset[i] of $\{0, \dots, n'\}$, and including before the first iteration (with $t = 1$), the algorithm maintains a function $y_t\colon V' \to \mathbb{R} \cup \{\infty\}$ and the variables $P_t(v)$ for each $v \in V'$ are such that (exercise)

$$\begin{aligned} &\text{either } \Big(P_t(v) = \perp \text{ and } y_t(v) = \infty\Big) \\ &\text{or } \Big(P_t(v) \text{ is an } rv\text{-path in } D' \text{ of cost } y_t(v) < \infty \text{ and length} \le t \Big). \end{aligned} \tag{4.10}$$

Also, if $t \ge 1$, the way the function y_{t-1} is updated to the next iteration is given by

$$y_t(v) = \min\Big(\big\{y_{t-1}(v)\big\} \cup \big\{ y_{t-1}(u) + c(a) : a \in \delta^{\text{in}}_{D'}(v),\, \varphi(a) = uv\big\}\Big), \qquad \text{for each } v \in V', \tag{4.11}$$

where we adopt the abuse of notation $\delta^{\text{in}}_D(v) \coloneqq \delta^{\text{in}}_D(\{v\})$ for each $v \in V$. See Figure 4.3. In particular,

$$y_t(v) \le y_{t-1}(v) \qquad \text{for each } v \in V', \tag{4.12}$$

and

$$\text{equality in (4.12) implies that } P_t(v) = P_{t-1}(v). \tag{4.13}$$

We will analyze several aspects of Algorithm 4.1 separately before combining our conclusions in Theorem 5.5 on the overall correctness of the algorithm.

See Figure 4.4 for an sample run.

Lecture 5: The Bellman-Ford Algorithm

5.1 Theorem. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. Suppose Algorithm 4.1 is run on input (D, c, r, s), let $v \in V'$ and let $t \in \{0, \dots, n'\}$ be such that $y_t(v)$ is defined. Then $y_t(v)$ is the optimal value of the optimization problem (SWP$_{\le t}(D', r, v)$), and $P_t(v)$ is an optimal solution whenever (SWP$_{\le t}(D', r, v)$) is feasible.

Proof. The proof is by induction on t, the base case $t = 0$ being true by the definition of y_0 and P_0 in Algorithm 4.1. Suppose that $t \ge 1$. By Theorem 1.5, the optimization problem (SWP$_{\le t}(D', r, v)$) is either infeasible or it has an optimal solution. Let $\mu^* \in \mathbb{R} \cup \{\infty\}$ be the optimal value of (SWP$_{\le t}(D', r, v)$). Then

$$\mu^* \le y_t(v) \tag{5.1}$$

follows from (4.10). In particular, equality holds if (SWP$_{\le t}(D', r, v)$) is infeasible.

---

[1] If you have seen the Bellman-Ford Algorithm before, most likely it had very little resemblance to Algorithm 4.1. There are several reasons for this.
One reason is that there are no *variables*, in the sense that, once a value has been assigned (or bound) to a symbol, this value never changes. In this sense, the "variables" behave more like mathematical symbols in a theorem, and this is the motivation for not having variables: we can safely refer to $y_4(v)$ for some vertex v, and not to "the value of $y(v)$ at iteration 4" (which raises several questions: at which point of the iteration? which iteration is iteration 4? do we start counting from 1 or 0?). Perhaps it would be more precise to say that there is no mutation (or side effects), rather than no variables. We aim to keep this style throughout this text. You may check the Haskell programming language if you are interested.
Another reason is that we keep track of extra information (one path for each vertex), which allows us to return a negative cycle almost immediately once the algorithm detects that such a cycle exists.
By dropping these characteristics, namely non-mutability and quick construction of negative cycle, it is possible to speed up the algorithm considerably, to $O(nm)$ time and $O(n)$ space.

[i] We cannot say that this holds for *every* index in the set $\{0, \dots, n'\}$ because one cannot know beforehand the number of iterations of the **for** loop in Line 14 that will be executed.

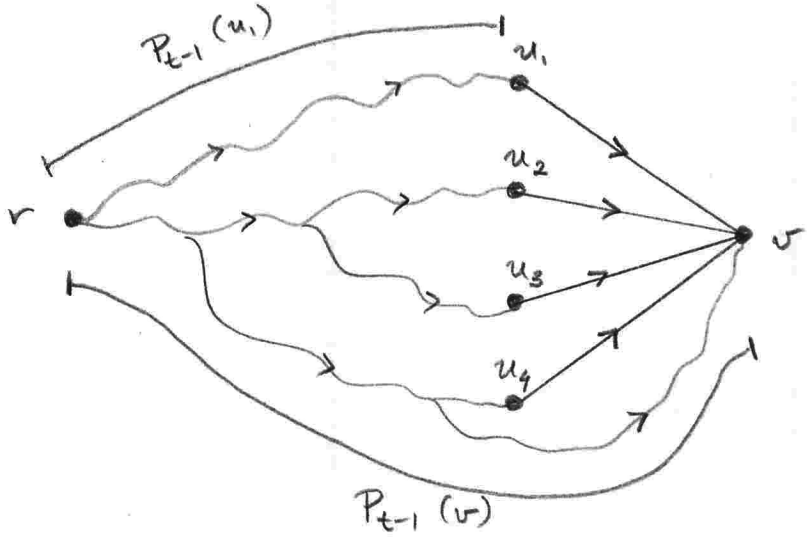


FIGURE 4.3. Illustration for the recurrence in Algorithm 4.1, using the notation from §4.12.

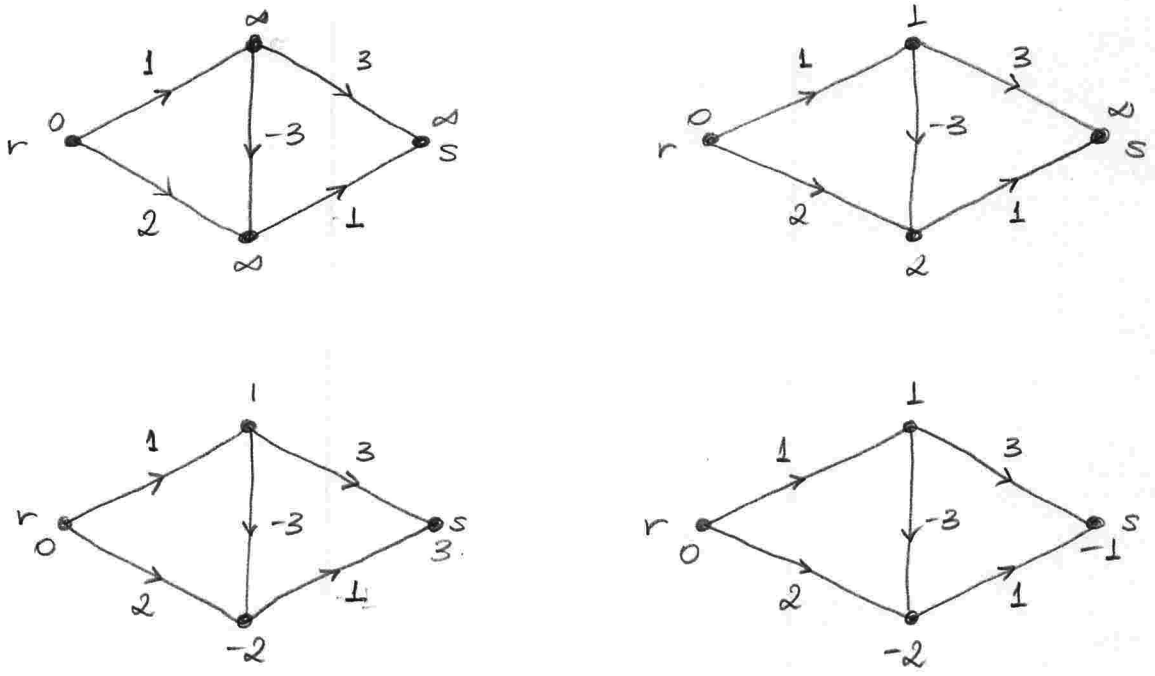


FIGURE 4.4. Sample run for the Bellman Ford Algorithm 4.1.

Suppose that (SWP$_{\leq t}(D', r, v)$) is feasible, so that μ^* is finite and there is an optimal solution $W^* := \langle v_0, \ldots, a_\ell, v_\ell \rangle$. If $\ell = 0$, then $v = r$ and

$$y_t(v) = y_t(r) \overset{(4.12)}{\leq} y_0(r) = 0 = c(W^*) = \mu^*,$$

so equality holds in (5.1). Thus, $P_t(v) = P_t(r) = P_0(r) = \langle r \rangle = W^*$ by (4.13).

Algorithm 4.1 The Bellman-Ford Algorithm for Problem 2.3

Input: a tuple (D, c, r, s), formed by
(i) a digraph $D = (V, A, \varphi)$,
(ii) a cost function $c\colon A \to \mathbb{R}$ on the arcs,
(iii) a source vertex $r \in V$ and a destination vertex $s \in V$.

Output: Either one of the following, each being an outcome for (SWP) and a certificate:
(i) infeasible, and a set $R \subseteq V$ such that $r \in R$, $s \notin R$, and $\delta^{\text{out}}_D(R) = \varnothing$, as in Corollary 3.13;
(ii) unbounded, a negative cycle C, an rv-path and a vs-path in D for some $v \in V(C)$, as in Corollary 3.4;
(iii) an rs-path P which is optimal for (SWP), sets of vertices R and S such that $r \in R$ and $\delta^{\text{out}}_D(R) = \varnothing$, $s \in S$ and $\delta^{\text{in}}_D(S) = \varnothing$, and a feasible potential y on $D[R \cap S]$ such that $c(P) = y(s) - y(r)$, as in Corollary 4.9.

1: **function** BELLMAN-FORD(D, c, r, s)
2: $R \leftarrow \{ v \in V : r \rightsquigarrow_D v \}$
3: $S \leftarrow \{ v \in V : v \rightsquigarrow_D s \}$
4: **if** $s \notin R$ **then**
5: **return** infeasible and R
6: $V' \leftarrow R \cap S$
7: $D' \leftarrow D[V'] = D[R \cap S]$
8: $n' \leftarrow |V'| = |R \cap S|$
9: $y_0(r) \leftarrow 0$
10: $P_0(r) \leftarrow \langle r \rangle$
11: **for all** $v \in V' \setminus \{r\}$ **do**
12: $y_0(v) \leftarrow \infty$
13: $P_0(v) \leftarrow \perp$ ▷ the symbol $\perp$ stands for "bottom"; you may think of it as NIL
▷ For each $v \in V'$ and t, either ($P_t(v) = \perp$ and $y_t(v) = \infty$), or $P_t(v)$ is an rv-path of cost $y_t(v) < \infty$
14: **for** $t \leftarrow 1$ **to** n' **do**
▷ we will set $y_t(v) = \min\Big(\{y_{t-1}(v)\} \cup \{\, y_{t-1}(u) + c(a) : a \in \delta^{\text{in}}_{D'}(v),\ \varphi(a) = uv \}\Big)$ for each $v \in V$
15: **for all** $v \in V'$ **do**
16: **if** $y_{t-1}(v) > \inf\{\, y_{t-1}(u) + c(a) : a \in \delta^{\text{in}}_{D'}(v),\ \varphi(a) = uv\}$ **then**
17: Let a attain the infimum in Line 16, and let $u \in V$ such that $\varphi(a) = uv$
▷ truth of **if** test in Line 16 implies $y_{t-1}(u) < \infty$ so $P_{t-1}(u)$ is an ru-path of cost $y_{t-1}(u)$
18: **if** $v \in V(P_{t-1}(u))$ **then**
19: $\langle v_0, \ldots, a_\ell, v_\ell \rangle \leftarrow P_{t-1}(u)$
20: Let $i \in \{0, \ldots, \ell\}$ such that $v_i = v$
21: **return** unbounded, an rv_i-path, a $v_i s$-path, and $C \coloneqq \langle v_i, \ldots, a_\ell, v_\ell \rangle \cdot \langle u, a, v \rangle$
22: **else**
23: $y_t(v) \leftarrow y_{t-1}(u) + c(a)$
24: $P_t(v) \leftarrow P_{t-1}(u) \cdot \langle u, a, v \rangle$ ▷ $P_t(v)$ is a path, by the failure of the test in Line 18
25: **else**
26: $y_t(v) \leftarrow y_{t-1}(v)$
27: $P_t(v) \leftarrow P_{t-1}(v)$
28: **return** the rs-path $P_{n'}(s)$, the sets R and S, and $y_{n'}\colon V' \to \mathbb{R}$

Suppose next that $\ell \geq 1$. Set $u \coloneqq v_{\ell-1}$ and $a \coloneqq a_\ell$. Then

$$W' \coloneqq \langle v_0, \ldots, a_{\ell-1}, v_{\ell-1} \rangle \text{ is an optimal solution for } (\text{SWP}_{\leq \ell-1}(D', r, u)) \tag{5.2}$$

by Lemma 4.11. Hence, by induction hypothesis since $\ell - 1 < \ell \leq t$,

$$c(W') = y_{\ell-1}(u). \tag{5.3}$$

Thus

$$\mu^* = c(W^*) = c(W') + c(a) = y_{\ell-1}(u) + c(a) \quad \text{by (5.3)}$$

$$\begin{aligned} &\geq y_\ell(v) && \text{by (4.11)} \\ &\geq y_t(v) && \text{by (4.12) since } \ell \leq t. \end{aligned}$$

Thus, equality holds in (5.1), and the statement about $P_t(v)$ follows from (4.10). □

5.2 Theorem (Negative Cycle Detection in the Bellman-Ford Algorithm). Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. Suppose that, upon running Algorithm 4.1 on input (D, c, r, s), the **if** condition in Line 18 holds true at some point. Then the sequence C returned in Line 21 is a negative cycle.

Proof. Execution of the **if** condition in Line 18 implies that Line 17 was just executed, and so the comment on this line shows that $y_{t-1}(u)$ is finite. Hence, by Theorem 5.1,

$$P_{t-1}(u) \text{ is an optimal solution for } (\mathrm{SWP}_{\leq t-1}(D', r, u)) \tag{5.4}$$

and

$$c\big(P_{t-1}(u)\big) = y_{t-1}(u). \tag{5.5}$$

Also, the **if** condition in Line 16 implies that

$$y_{t-1}(v) > y_{t-1}(u) + c(a). \tag{5.6}$$

Following the notation from Line 19, we have $P_{t-1}(u) = \langle v_0, \ldots, a_\ell, v_\ell\rangle$, and

$$v = v_i. \tag{5.7}$$

Since $P_{t-1}(u)$ is a path by (4.10), so is $W_{i\ell} \coloneqq \langle v_i, \ldots, a_\ell, v_\ell\rangle$. Thus,

$$C = \langle v_i, \ldots, a_\ell, v_\ell\rangle \cdot \langle u, a, v\rangle = W_{i\ell} \cdot \langle u, a, v\rangle \tag{5.8}$$

is a cycle.

Set $W_{0i} \coloneqq \langle v_0, \ldots, a_i, v_i\rangle$, so that $P_{t-1}(u) = W_{0i} \cdot W_{i\ell}$ and

$$c\big(P_{t-1}(u)\big) = c(W_{0i}) + c(W_{i\ell}). \tag{5.9}$$

By (5.4) and Lemma 4.11, the walk $W_{0i} = \langle v_0, \ldots, a_i, v_i\rangle$ is optimal for $(\mathrm{SWP}_{\leq i}(D', r, v_i))$, whence

$$y_i(v_i) = c(W_{0i}) \tag{5.10}$$

follows from Theorem 5.1.

The cost of the cycle C is

$$\begin{aligned} c(C) &= c(W_{i\ell}) + c(a) && \text{by (5.8)} \\ &= c\big(P_{t-1}(u)\big) - c\big(W_{0i}\big) + c(a) && \text{by (5.9)} \\ &= y_{t-1}(u) - y_i(v_i) + c(a) && \text{by (5.5) and (5.10)} \\ &= y_{t-1}(u) + c(a) - y_i(v) && \text{by (5.7)} \\ &< y_{t-1}(v) - y_i(v) && \text{by (5.6)} \\ &\leq 0 && \text{by (4.12) since } i \leq \ell \leq t-1. \end{aligned}$$

That is, C is a negative cycle. □

5.3 Exercise. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r \in V$. Let $y\colon V \to \mathbb{R}$, and define $z\colon V \to \mathbb{R}$ by setting

$$z(v) \coloneqq \min\Big(\big\{y(v)\big\} \cup \big\{\, y(u) + c(a) : a \in \delta^{\mathrm{in}}_D(v),\, \varphi(a) = uv\big\}\Big), \qquad \text{for each } v \in V. \tag{5.11}$$

Prove that y is a feasible potential if and only if $z = y$.

5.4 Theorem (Feasible Potentials at the End of the Bellman-Ford Algorithm). Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. Suppose that, upon running Algorithm 4.1 on input (D, c, r, s), the **if** condition in Line 18 never holds true, so the algorithm terminates upon execution of the **return** in Line 28. Then $y^* \coloneqq y_{n'}$ is a feasible potential for $D' = D[R \cap S]$.

Proof. We first note that

$$y^*(v) < \infty \qquad \text{for each } v \in R \cap S. \tag{5.12}$$

Let $v \in R \cap S$. Then $r \rightsquigarrow_{D'} v$ by Exercise 4.8. Thus, there is an rv-walk in D', and hence there is an rv-path in D' by Corollary 3.2. Since every path in D' has length $\leq n' - 1$, where $n' = |V(D')|$, the optimization problem (SWP$_{\leq n'-1}(D', r, v)$) is feasible and so $y_{n'-1}(v) < \infty$ by Theorem 5.1. Thus, $y^*(v) = y_{n'}(v) \leq y_{n'-1}(v) < \infty$, by using (4.12). This proves (5.12).

Next, to prove that y^* is a feasible potential for D', by Exercise 5.3, it suffices to show that

$$y_{n'} = y_{n'-1}. \tag{5.13}$$

Suppose not, for the sake of contradiction. Then the condition of the **if** in Line 16 must have been tested positive for some $v \in V'$ and $t = n'$, thus yielding $y_{n'}(v) < y_{n'-1}(v)$. We claim that

(5.14) the test condition of the **if** in Line 18 in this iteration must be true.

If not, then the **else** in Line 22 is run, so that $P_{n'}(v)$ is a path that is an optimal solution to (SWP$_{\leq n'}(D', r, v)$) by Theorem 5.1, with objective value $y_{n'}(v) < y_{n'-1}(v)$. The latter is, by Theorem 5.1, a lower bound on the optimal value of the shortest **path** problem from r to v in D'. So

$$c(P_{n'}(v)) = y_{n'}(v) < y_{n'-1}(v) \leq \min\{\, c(P) : P \text{ an } rv\text{-path in } D'\},$$

a contradiction. This proves (5.14).

However, (5.14) contradicts our assumption that the **if** condition in Line 18 never holds true. Thus, (5.13) holds, and $y^* = y_{n'}$ is a feasible potential for D' by Exercise 5.3. □

5.5 Theorem (Correctness of the Bellman-Ford Algorithm 4.1). Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. Suppose Algorithm 4.1 is run on input (D, c, r, s). Then precisely one of the following holds[1]:

(5.15)
- (i) (SWP) is infeasible, and Line 5 returns a subset $R \subseteq V$ such that $r \in R$, $s \notin R$, and $\delta^{\text{out}}_D(R) = \varnothing$, i.e., the conditions from Corollary 3.13;
- (ii) (SWP) is unbounded, and Line 21 returns, for some vertex $v \in V$, an rv-path P_r, a vs-path P_s, and a cycle C such that $v \in V(C)$ and $c(C) < 0$, i.e., the conditions from Corollary 3.4;
- (iii) (SWP) has an optimal solution, and Line 28 returns an rs-path P, sets $R, S \subseteq V$ and a function $y^*\colon R \cap S \to \mathbb{R}$ such that
  - (a) P is an optimal solution for (SWP),
  - (b) $r \in R$ and $\delta^{\text{out}}_D(R) = \varnothing$,
  - (c) $s \in S$ and $\delta^{\text{in}}_D(S) = \varnothing$,
  - (d) y^* is a feasible potential for $D[R \cap S]$, and
  - (e) $c(P) = y^*(s) - y^*(r)$,

  i.e., the conditions from Corollary 4.9.

Proof. That at least one of the conditions from (5.15) holds follows from executing the algorithm, and it is obvious that they are mutually exclusive. It is also clear that (5.15)(i) holds if and only if the test condition in **if** from Line 4 holds true.

Theorem 5.2 shows that, if the test condition of the **if** in Line 18 ever holds true, then the cycle C returned by Line 21 is a negative cycle that satisfies the conditions from Corollary 3.4.

So suppose that the test condition of the **if** in Line 18 always fails. Then $y^* := y_{n'} \in \mathbb{R}^{R \cap S}$ is a feasible potential for $D[R \cap S]$ by Theorem 5.4. Hence, $D[R \cap S]$ has no negative cycles by Corollary 4.5. Thus, by Theorem 5.1, we have

$$y^*(r) = y_{n'}(r) = 0. \tag{5.16}$$

The path $P := P_{n'}(s)$, returned by Line 28 is an rs-path of cost $y_{n'}(s) = y^*(s)$ by (4.10), so $c(P) = y^*(s) = y^*(s) - y^*(r)$ by (5.16). The set R returned by Line 28 satisfies $r \in R$ and $\delta^{\text{out}}_D(R) = \varnothing$ by Exercise 3.12. A symmetric argument applied to D' and s shows that $s \in S$ and $\delta^{\text{in}}_D(S) = \varnothing$. □

[1] You may view this result as a refinement of Corollary 3.7, in which not only short (polynomial sized) certificates are provided for each outcome, but also everything is algorithmic and efficient.

5.6 Corollary. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function. Then there is no negative cycle in D if and only if there is a feasible potential for D.

Proof. Sufficiency: This is Corollary 4.5.

Necessity: Suppose there is no negative cycle. Let r, s be *new* vertices[i], i.e., elements not in V. Intuitively, we will form a new digraph[1] with vertex set $V \cup \{r, s\}$, all original arcs from D, plus new arcs of the form rv and vs for each $v \in V$, each with cost 0. See Figure 5.1. Set $V' \coloneqq V \cup \{r, s\}$. Set $A_r \coloneqq \{\, rv : v \in V \,\}$ and $A_s \coloneqq \{\, vs : v \in V \,\}$. Set[2]

$$A' \coloneqq A_r \sqcup A \sqcup A_s = (\{1\} \times A_r) \cup (\{2\} \times A) \cup (\{3\} \times A_s) \tag{5.17}$$

and define $\varphi'\colon A' \to \mathbb{R}$ by setting

$$\varphi'((i,a)) \coloneqq \begin{cases} \varphi(a) & \text{if } i = 2, \\ a & \text{otherwise.} \end{cases}$$

Next, set $D' \coloneqq (V', A', \varphi')$, and define the cost function[3]

$$c'((i,a)) = [i = 2]c(a) \qquad \text{for each } (i,a) \in A'$$

Since $\delta^{\text{in}}_{D'}(r) = \varnothing = \delta^{\text{out}}_{D'}(s)$, every cycle C in D' has $V(C) \subseteq V' \setminus \{r, s\} = V$, i.e., every cycle in D' is a cycle in D. In particular, D' has no negative cycle, and clearly $r \rightsquigarrow_{D'} s$. Hence, by Theorem 5.5 applied to[ii] (D', c', r, s), there are subsets $R, S \subseteq V'$ and a function $y^*\colon R \cap S \to \mathbb{R}$ such that $r \in R$ and $\delta^{\text{out}}_{D'}(R) = \varnothing$, $s \in S$ and $\delta^{\text{in}}_{D'}(S) = \varnothing$, and y^* is a feasible potential for $D'[R \cap S]$. By the choice of the arcs in A_r, we have $V \subseteq R$. Similarly, using the arcs in A_s, we have $V \subseteq S$. Thus, the restriction of $y^*\colon R \cap S \to \mathbb{R}$ to $V \subseteq R \cap S$ is a feasible potential for $D'[V] = D$. □

Lecture 6: Optimization Homomorphisms and Graphs

6.1 Level Sets. Let $(X, f, \alpha) =: \mathcal{O}$ be an optimization problem, and let $\mu \in \mathbb{R}$. The **μ-level set** of $\mathcal{O}$ is

$$L_{\mathcal{O}}(\mu) \coloneqq \{\, x \in X : \alpha f(x) \geq \alpha\mu \,\}, \tag{6.1}$$

i.e.,

$$L_{\mathcal{O}}(\mu) = \begin{cases} \{\, x \in X : f(x) \geq \mu \,\} = f^{-1}\big([\mu, +\infty)\big) & \text{if } \alpha = +1, \\ \{\, x \in X : f(x) \leq \mu \,\} = f^{-1}\big((-\infty, \mu]\big) & \text{if } \alpha = -1. \end{cases}$$

Here we using the notation of preimage: if $g\colon A \to B$ is a function and $C \subseteq B$, then the **preimage of C under g** is

$$g^{-1}(C) \coloneqq \{\, a \in A : g(a) \in C \,\}, \tag{6.2}$$

that is, $g^{-1}(C)$ is the set of all domain points which are mapped to C by g. Note that this notation does *not* imply (!) that g is invertible; and it can be used for *any* function. (Of course, if g is invertible, then one

[1] We will reduce the problem of finding a feasible potential for this digraph D to another problem over a slightly different digraph (which *we* build, and over which we have more control), constructed from D. This will be a recurring theme throughout the course.

[2] Here we are using the construction of disjoint sets. We are given three sets, A_r, A, and A_s, and we want to take their union, while making sure that, if some element inconspicuously is common to both A_r and A, they yield different elements in the *disjoint* union. This is required because we have absolutely no control over the set A, which may be provided to us by our enemy. A standard construction to achieve that is to, literally, *tag* each element to know from which set it came from. That is, the disjoint union, denoted by $A_r \sqcup A \sqcup A_s$, is defined as in (5.17). More generally, given sets $S_1, \ldots, S_k$, the disjoint union of these sets is $S_1 \sqcup \cdots \sqcup S_k \coloneqq \bigcup_{i=1}^{k} \{i\} \times S_i$.

[3] Here we start employing the extremely versatile Iverson notation. Given a predicate P, i.e., an expression that evaluates to either true or false, $[P]$ is defined to be 1 if the predicate P is true, and zero otherwise. In fact, it is convenient to think of the false/zero case as being "strongly zero", in the sense that, whatever comes after $[P]$ *need not even parse* if P is false, and still the value of the whole expression is taken to be *zero*. For example, the expression $[x \neq 0](1/x)$ is valid, and when $x = 0$ the expression evaluates to 0, even though the expression that follows the Iverson bracket does not parse if $x = 0$. Think of short circuiting and boolean conversions in the C programming language.

[i] It is a basic result in set theory that, given any set S, there is an element (which is a set, because, in mathematics, what isn't?) which does not lie in S.

[ii] As in the proof of Corollary 3.2, note the we rely on the **statement** of Theorem 5.5, not its proof. That is, we are using the result as a black box, again providing proper encapsulation and treating theorem statements as APIs.

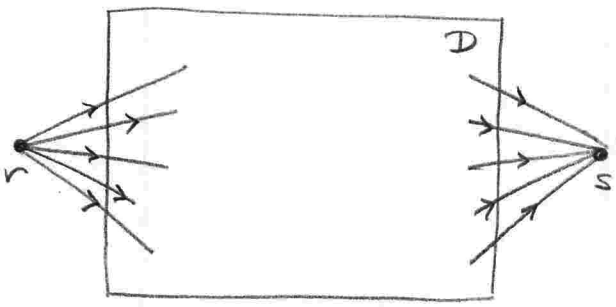


FIGURE 5.1. Constructing the digraph D' from the input digraph D as in the proof of Corollary 5.6.

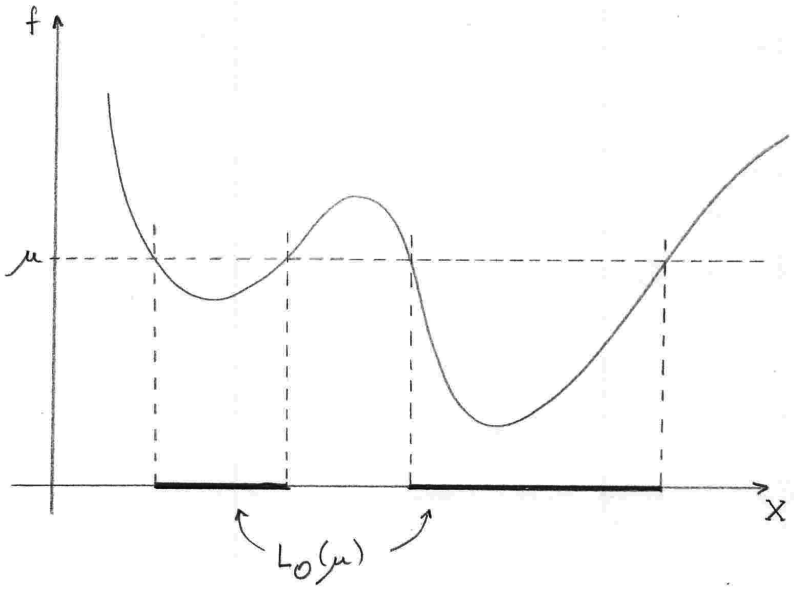


FIGURE 6.1. Illustration of a level set.

may still use g^{-1} to denote its inverse, as long as the context makes it clear which concept the overloaded notation intends to use.)

See Figure 6.1.

6.2 Homomorphisms of Optimization Problems. Let $\mathcal{O} := (X, f, \alpha)$ and $\mathcal{P} := (Y, g, \alpha)$ be optimization problems[i]. A(n optimization problem) **homomorphism** from $\mathcal{O}$ to $\mathcal{P}$ is a function $\varphi\colon X \to Y$ such that,

$$\varphi\big(L_{\mathcal{O}}(\mu)\big) \subseteq L_{\mathcal{P}}(\mu) \qquad \text{for each } \mu \in \mathbb{R}, \tag{6.3}$$

that is, φ maps level sets of $\mathcal{O}$ to (the corresponding) level sets of $\mathcal{P}$. It is a simple exercise to prove that a function

$$\varphi\colon X \to Y \text{ is a homomorphism from } \mathcal{O} \text{ to } \mathcal{P} \iff \text{for each } x \in X,\ \begin{cases} g(\varphi(x)) \geq f(x) & \text{if } \alpha = +1, \\ g(\varphi(x)) \leq f(x) & \text{if } \alpha = -1. \end{cases} \tag{6.4}$$

[i]Note that we demand that $\mathcal{O}$ and $\mathcal{P}$ are either both maximization or both minimization problems.

In other words, a homomorphism sends feasible solutions to feasible solutions, and it can only improve the objective values. (We will use (6.4) interchangeably with the definition of homomorphism without further mention.)

We write $\mathcal{O} \xrightarrow{\varphi} \mathcal{P}$ to denote that φ is an homomorphism from $\mathcal{O}$ to $\mathcal{P}$, and we just write $\mathcal{O} \to \mathcal{P}$ if we do not care to name the homomorphism. It should be simple to verify that

$$\text{if } \mathcal{O} \to \mathcal{P}, \text{ then } \begin{cases} \mathrm{OPT}(\mathcal{O}) \leq \mathrm{OPT}(\mathcal{P}) & \text{if } \alpha = +1, \\ \mathrm{OPT}(\mathcal{O}) \geq \mathrm{OPT}(\mathcal{P}) & \text{if } \alpha = -1. \end{cases} \tag{6.5}$$

6.3 Examples of Homomorphisms. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. Consider the problems (SWP) and (SPP). Since every rs-path is an rs-walk (and the objective functions in both problems are the same), the identity function is a homomorphism from (SPP) to (SWP), and in fact the (bottom) inequality in (6.4) holds with equality.

For another example with the same context, suppose there are no negative cycles. Then Corollary 3.6[1] shows that for every feasible solution for (SWP) there is a feasible solution for (SPP) with at least as good objective value, so there is a homomorphism from (SWP) to (SPP). Note also that such a homomorphism is built from Theorem 3.1, which is algorithmic[i], i.e., the proof of Theorem 3.1 provides an algorithm that, given an rs-walk W in D, builds an rs-path P of cost $c(P) \leq c(W)$. Note that this inequality may be strict[ii] in this case.

6.4 Isomorphisms and Homomorphic Equivalence. Let $\mathcal{O} \coloneqq (X, f, \alpha)$ and $\mathcal{P} \coloneqq (Y, g, \alpha)$ be optimization problems. If φ is a homomorphism from $\mathcal{O}$ to $\mathcal{P}$ such that φ is a bijection and its inverse $\varphi^{-1}\colon Y \to X$ is a homomorphism from $\mathcal{P}$ to $\mathcal{O}$, then we call φ an (optimization problem) **isomorphism**, and problems $\mathcal{O}$ and $\mathcal{P}$ are **isomorphic**.

From previous experience with the term "isomorphism", you can probably realize that, if $\mathcal{O}$ and $\mathcal{P}$ are isomorphic, then "$\mathcal{O}$ and $\mathcal{P}$ are the same, in everything that matters". That is to say, the feasible sets are in bijective correspondence and the objective values are the same, so the only thing that may possibly be different in $\mathcal{O}$ and $\mathcal{P}$ are the names/labels of the feasible solutions. With this interpretation in mind, it should not be surprising that the outcomes of two isomorphic optimization problems are the same, and that from an optimal solution of one of the problems (if this is the outcome), one may obtain an optimal solution of the other by using the isomorphism. However, the same property may be obtained from a weaker notion, which we now introduce.

Let $\mathcal{O}$ and $\mathcal{P}$ be optimization problems. If $\mathcal{O} \to \mathcal{P}$ and $\mathcal{P} \to \mathcal{O}$, we say that $\mathcal{O}$ and $\mathcal{P}$ are **(homomorphically) equivalent**.

6.5 Exercise. Let $\mathcal{O}$ and $\mathcal{P}$ be equivalent optimization problems. Prove that $\mathrm{OPT}(\mathcal{O}) = \mathrm{OPT}(\mathcal{P})$ and that precisely one of the following holds:

(6.6)
- (i) $\mathcal{O}$ and $\mathcal{P}$ are infeasible;
- (ii) $\mathcal{O}$ and $\mathcal{P}$ are unbounded;
- (iii) $\mathcal{O}$ and $\mathcal{P}$ have (the same) finite optimal values and no optimal solutions;
- (iv) $\mathcal{O}$ and $\mathcal{P}$ have optimal solutions; moreover, if $\mathcal{O} \xrightarrow{\varphi} \mathcal{P}$ and x^* is an optimal solution for $\mathcal{O}$, then $\varphi(x^*)$ is an optimal solution for $\mathcal{P}$.

6.6 In §6.3, we discussed that the shortest walk problem and the shortest path problem, for the same instance, are equivalent. The proof of Corollary 3.7, we mentioned informally that those problems are "equivalent". Now we have a rigorous definition of this term, and Exercise 6.5 justifies the arguments made in that proof.

---

[1] A previous version of these notes had this fact as an intermediate result in the middle of the proof of Corollary 3.7. To make proper encapsulate/modularize our "code", the current version splits that intermediate result as a self-contained result, which can now be properly "exported" as part of our API.

[i] That is to say, we are not just stating an abstract existence result of some intangible homomorphism, but rather such homomorphism is so concrete that we can code it up as an algorithm.

[ii] It is quite common to, upon seeing the definition of homomorphism for the first time, to ask whether we need the inequalities in (6.4). This example illustrates why we allow the inequality.

More importantly for us, the homomorphisms described in §6.3 are algorithmic. This is crucial for us, since we are not only interested in determining an outcome, but also in finding optimal solutions when they exist, and in doing so in an algorithmically efficient manner[i].

6.7 Graphs. For a set V and a number $k \in \mathbb{N}$, denote by $\binom{V}{k}$ the collection of all k-**subsets** of V, that is, all subsets of V of size k.

A **graph**[ii] is a triple $(V, E, \varphi) =: G$ where

(i) V is a set, called the **vertex**[1] **set** of G,
(ii) E is a set, called the **edge**[2] **set** of G, and
(iii) $\varphi\colon E \to \binom{V}{1} \cup \binom{V}{2}$ is a function, called the **incidence function** of G.

If $e \in E$ is an edge and $\varphi(e) = \{u, v\}$ for some $u, v \in V$, then

(i) u and v are the **ends** of e,
(ii) u and v are **adjacent** or **neighbors** (in G),
(iii) u and v are **incident** with e.

Whenever it is clear from context, we may abbreviate the unordered pair[3] $\{u, v\}$ as uv. If $|\varphi(e)| = 1$ then e is a **loop**; otherwise e is a **non-loop**. A graph without loops is **loopless**. If $e, f \in E$ are non-loops such that $\varphi(e) = \varphi(f)$, the edges e and f are **parallel**.

For a graph G, we usually denote the vertex set of G by $V(G)$ or V_G, and the edge set of G by $E(G)$ or E_G. Throughout this text,

$$\text{all graphs are } \textbf{finite}, \tag{6.7}$$

that is, the vertex set and the edge set are both finite.

Whenever the incidence function is the identity function, we may omit φ from the triple $G = (V, E, \varphi)$ and we may just write $G = (V, E)$.

The **degree**[iii] of a vertex $v \in V$ is

$$\deg_G(v) \coloneqq |\{\, e \in E : e \text{ a non-loop and } v \text{ incident with } e\}| + 2|\{\, e \in E : e \text{ a loop and } v \text{ incident with } e\}|.$$

The **minimum** and **maximum degrees** of G are, respectively,

$$\delta(G) \coloneqq \min_{v \in V} \deg_G(v) \qquad \text{and} \qquad \Delta(G) \coloneqq \max_{v \in V} \deg_G(v).$$

If all vertices of G are equal to $k \in \mathbb{N}$, we say that G is k-**regular**. In this case, k is the **valency** of G. When the valency is not relevant, we may just say that G is **regular**.

6.8 Walks. A **walk** in a graph $G \coloneqq (V, E, \varphi)$ is a finite sequence of the form $\langle v_0, e_1, v_1, \ldots, e_\ell, v_\ell \rangle =: W$, where

(i) $\ell \in \mathbb{N}$ is the **length** of W,
(ii) $v_0, \ldots, v_\ell \in V$, and
(iii) $e_1, \ldots, e_\ell \in E$ are such that $\varphi(e_i) = \{v_{i-1}, v_i\}$ for each $i \in [\ell]$.

---

[1] Therefore, we call each element of V a **vertex** of D.
[2] Therefore, we call each element of E an **edge** of G.
[3] Hence, when you see uv in this text, it might mean either the product $u \cdot v$, the ordered pair (u, v) as in §2.1, or the unordered pair $\{u, v\}$. The context will make it clear which one is meant, i.e., it will be the only one that parses.

[i] Indeed, the notion of homomorphic equivalence between optimization problems (and in fact, even the notion of a homomorphism between optimization problems) is almost a triviality from the point of view of "pure" mathematics. The reason is that Exercise 6.5 can be refined to be an "if and only if", in the following sense. If $\mathcal{O}$ and $\mathcal{P}$ are optimization problems (with the same sign of α) with the same outcome and same optimal value, it can be proved that $\mathcal{O}$ and $\mathcal{P}$ are homomorphically equivalent. Thus, huge swaths of optimization problems are equivalent, even if they have nothing to do with one another except for outcome and optimal value.
However, our interest in such homomorphisms in this text is that we want our homomorphisms to be efficient algorithms, which enables translating optimal solutions between distinct problems as in (6.6)(iv), for instance.
[ii] Similarly as for digraphs, the term "undirected graph" is also quite common. Again, since we shall use the term often in this text, we prefer to just call it a "graph".
[iii] The motivation for making a loop contribute two units to the degree of its endpoint is to ensure that the "classical identity" $\sum_{v \in V} \deg_G(v) = 2|E_G|$ still holds, even when there are loops.

We denote $V(W) \coloneqq \{v_0, \ldots, v_\ell\}$ and $E(W) \coloneqq \{e_1, \ldots, e_\ell\}$. We say that W is a walk **from** v_0 **to** v_ℓ, or a (v_0, v_ℓ)**-walk**. The walk is called **even** if the length ℓ is even and **odd** if the length ℓ is odd.

The walk W is[i]:

(i) **closed** if $\ell \geq 1$ and $v_0 = v_\ell$,
(ii) a **trail** if all of its edges are distinct, i.e., the map $i \in [\ell] \mapsto a_i$ is injective,
(iii) a **path** if all of its vertices are distinct, i.e., the map $i \in \{0, \ldots, \ell\} \mapsto v_i$ is injective,
(iv) a **cycle** if it is a closed trail and the map $i \in [\ell] \mapsto v_i$ is injective.

A graph G is **acyclic** if there are no cycles in G. An acyclic graph is also called a **forest**.

As for digraphs, if $W \coloneqq \langle v_0, e_1, v_1, \ldots, e_\ell, v_\ell \rangle$ is a walk from v_0 to v_ℓ and $Z \coloneqq \langle u_0, f_1, u_1, \ldots, f_k, u_k \rangle$ is a walk from $u_0 = v_\ell$ to u_k, we denote by

$$WZ \coloneqq W \cdot Z \coloneqq \langle v_0, e_1, v_1, \ldots, e_\ell, \underbrace{v_\ell}_{\| \, u_0}, f_1, u_1, \ldots, f_k, u_k \rangle \tag{6.8}$$

the concatenation of W and Z, with the repetition of $v_\ell = u_0$ dropped. So WZ is a walk from v_0 to u_k.

6.9 Subgraphs. A graph $G' \coloneqq (V', E', \varphi')$ is a **subgraph** of a graph $G \coloneqq (V, E, \varphi)$ if $V' \subseteq V$, $E' \subseteq E$, and $\varphi' = \varphi\restriction_{E'}$. In this case, we say that G is a **supergraph** of G'. The subgraph G' of G is **spanning** if $V' = V$.

Let $U \subseteq V$. We say that an edge $e \in E$ is **induced** by U if $\varphi(e) \subseteq U$, i.e., if both its ends lie in U. The set of all edges in E induced by U is denoted by $E_\varphi[U]$ or $E[U]$. The **subgraph** of G **induced by** U is $G[U] \coloneqq (U, E_\varphi[U], \varphi\restriction_{E_\varphi[U]})$.

6.10 Equivalence Relations, Equivalence Classes, and Partitions. Recall that a binary relation $\sim$ on a set V is a subset of the Cartesian product $V \times V$, i.e., $\sim$ is a set of ordered pairs (u, v) with $u, v \in V$. In this case, it is conventional to write $(u, v) \in \sim$ as $u \sim v$. The binary relation $\sim$ is an **equivalence relation** if

(i) $\sim$ is **reflexive**, i.e., $v \sim v$ for each $v \in V$;
(ii) $\sim$ is **symmetric**, i.e., $u \sim v$ implies $v \sim u$, for each $u, v \in V$;
(iii) $\sim$ is **transitive**, i.e., $u \sim v$ and $v \sim w$ imply that $u \sim w$, for each $u, v, w \in V$.

The **power set** of a set V is the collection of all subsets of V, and it is denoted by $\mathcal{P}(V)$. Sets X and Y are **disjoint** if $X \cap Y = \varnothing$. A collection $\mathcal{Q} \subseteq \mathcal{P}(V)$ of subsets of V is **(pairwise) disjoint** if any two distinct elements of $\mathcal{Q}$ are disjoint. A **partition** of V is a pairwise disjoint collection $\mathcal{Q} \subseteq \mathcal{P}(V) \setminus \{\varnothing\}$ of *nonempty* subsets of V such that $\bigcup \mathcal{Q} = V$. In this case, each element of $\mathcal{Q}$ is called a **part** or **block** of the partition[ii]

It is a standard (and straightforward) exercise to verify that, if $\sim$ is an equivalence relation on V, then the collection

$$V/\!\sim \coloneqq \Big\{ \{ u \in V : u \sim v \} : v \in V \Big\} \subseteq \mathcal{P}(V) \tag{6.9}$$

is a partition of V, each of whose blocks is called an **equivalence class** of $\sim$. Note that, in (6.9), for each $v \in V$, the block $\{ u \in V : u \sim v \}$ of $V/\!\sim$ is the set of all elements that are related to v under $\sim$ (normally read as "equivalent to v"), also called *the* **equivalence class of** v (under $\sim$).

6.11 Reachability and Components. Let $G \coloneqq (V, E, \varphi)$ be a graph. For $u, v \in V$, write $u \sim_G v$ if there is a walk from u to v in G. Note that $\sim_G$ is a binary relation on V, and it is straightforward to prove that $\sim_G$ is an equivalence relation. The equivalence classes of $\sim_G$ are called the **(connected) components** of G. The graph G is **disconnected** if it has at least two components, and G is **connected** otherwise. See Figure 6.2.

By our definition, a connected component of G is a subset of V. However, it is very convenient to *identify* a component $C \subseteq V$ of G with the induced subgraph $G[C]$, and we shall do so. By this convention, we may use the same symbol in a sentence in multiple/conflicting ways, e.g., we may say "the component C is connected[iii] and there exist distinct $u, v \in C$[iv]".

A graph that is connected and acyclic is called a **tree**. Verify that each component of a forest is a tree, which explains its name.

---

[i] All definitions enumerated are pairwise independent.
[ii] It is a common mistake to refer to the elements of a partition itself as a "partition". Train yourself not to do this.
[iii] So C is interpreted as an induced subgraph for this to parse.
[iv] So C is a set of vertices for this to parse.

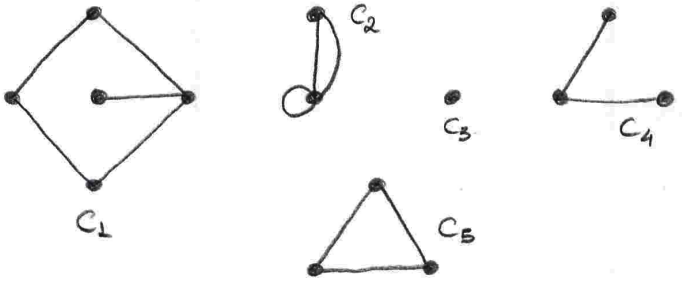
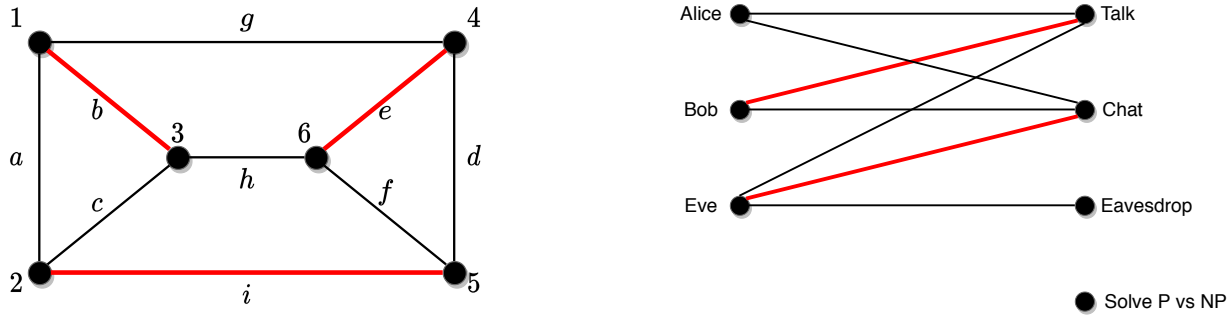


FIGURE 6.2. Example of connected components of a graph.



(A) (B)

FIGURE 7.1. The red edges in both graphs above are matchings, e.g., the set $\{b, e, i\}$ is a matching in the graph shown in Figure 7.1a.

LECTURE 7: MAXIMUM MATCHINGS

7.1 Matchings. Let $G \coloneqq (V, E, \varphi)$ be a graph. A set $M \subseteq E$ of edges is a **matching** if no edge of M is a loop and each vertex of G is incident with at most one edge in M. See Figure 7.1. Suppose M is a matching. A vertex of G that is incident with some edge in M is called **M-saturated**[i]. Denote the set of M-saturated vertices by V_M; we also say that M **saturates** the vertices in V_M. Vertices that are not M-saturated, i.e., vertices in $V \setminus V_M$ are **M-free**. The matching M is **perfect** if M saturates every vertex of G.

7.2 Problem (The Maximum Matching Problem). The **maximum matching problem** is, **given** a graph $G \coloneqq (V, E, \varphi)$, solve the optimization problem

$$\begin{aligned} &\text{Maximize} && |M| \\ &\text{subject to} && M \text{ is a matching in } G. \end{aligned} \tag{7.1}$$

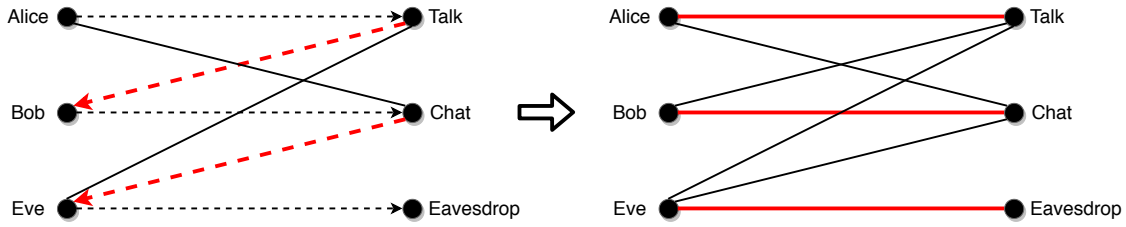
Note that (7.1) is feasible for every input graph G, since $\varnothing$ is a feasible solution. Thus,

(7.2) (7.1) has an optimal solution

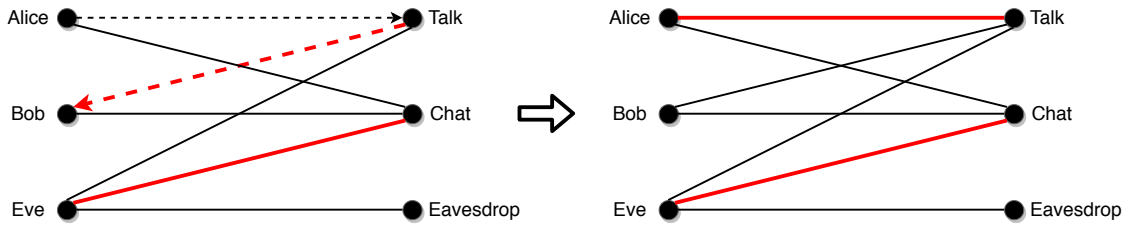
by Theorem 1.5. The **matching number** of G, denoted by $\nu(G)$, is the optimal value of (7.1). A matching of G of size $\nu(G)$ is called **maximum**.

Any polynomial-time algorithm for solving the maximum matching problem implies a polynomial-time algorithm for solving the **perfect matching problem**: **given** a graph $G \coloneqq (V, E, \varphi)$, decide if G has a perfect matching, and if so, find one.

[i] It is common to call such vertices M-covered or covered by M. However, we will avoid doing so, since the overloaded "cover" terminology will be used with different meaning in the next lectures and it may confuse readers.



(A) An M-augmenting path.



(B) An even M-alternating path.

FIGURE 7.2. Examples of M-alternating paths, and the "switched" matching $M \Delta E(P)$.

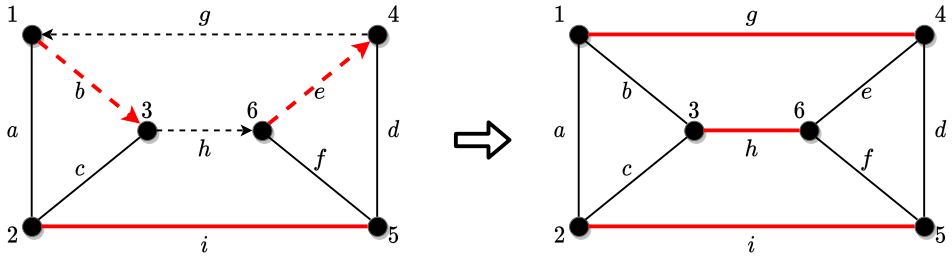


FIGURE 7.3. An M-alternating cycle $C := \langle 1, b, 3, h, 6, e, 4, g, 1\rangle$, and the "switched" matching $M \Delta E(C)$.

7.3 Alternating Paths and Cycles. Let M be a matching in a graph $G := (V, E, \varphi)$. Let $W := \langle v_0, \ldots, e_\ell, v_\ell\rangle$ be a walk in G that is a path or an even cycle. Set $r := v_0$ and $s := v_\ell$. We say that W is **M-alternating** if

(7.3) (i) the edges of W are alternately in and out of M, i.e., $e_i \in M \iff e_{i+1} \notin M$ for each $i \in [\ell - 1]$, and

(ii) for each $v \in \{r, s\}$, if v is M-saturated and v is incident with $e \in M$, then $e \in E(W)$.

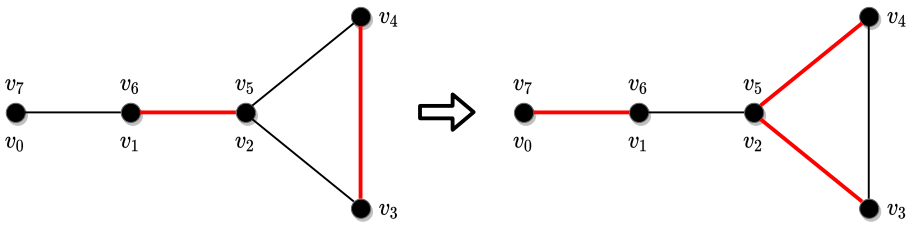
Suppose that W is M-alternating. Verify that $M' := M \Delta E(W)$ is a matching, and that

$$|M'| = \begin{cases} |M| + 1 & \text{if } W \text{ is an odd path with } r, s \notin V_M, \\ |M| & \text{if } W \text{ is an even path or an even cycle,} \\ |M| - 1 & \text{if } W \text{ is an odd path with } r, s \in V_M. \end{cases} \tag{7.4}$$

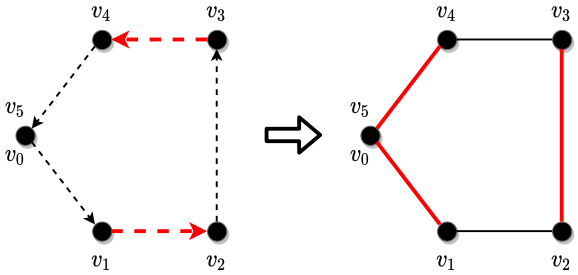
See Figures 7.2 and 7.3 for examples, as well as Figures 7.4 and 7.5 to understand the constraints on an M-alternating walk. Note that the cases in (7.4) exhaust all possibilities. Thus, we say that W is **M-augmenting** if $\ell \geq 1$ and $r, s \notin V_M$.

Clearly, if M is a matching and P is an M-augmenting path, then $M \Delta E(P)$ is a matching larger than M by (7.4), whence M is not optimal.

7.4 Paths and Cycles as Graphs. A graph $G := (V, E, \varphi)$ is a **path** if there is a path P in G such that $G = (V(P), E(P), \varphi)$. Similarly, G is a **cycle** if there is a cycle C in G such that $G = (V(C), E(C), \varphi)$. In both cases, we are overloading the meaning of paths and cycles, so these terms may refer either to a walk



(A) Repeated "inner" vertices are not allowed in M-alternating walks.



(B) Odd cycles are not M-alternating.

FIGURE 7.4. These examples illustrate why we require M-alternating walks to have no repeated vertices, except for the first and last vertices, and in this case we only allow even cycles.

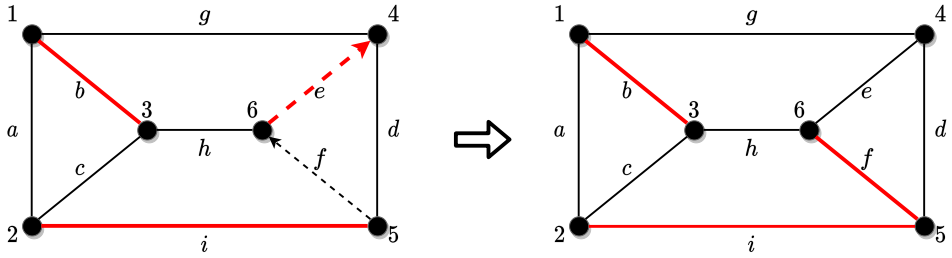


FIGURE 7.5. This example illustrates why we require the "technical" condition (7.3)(ii).

FIGURE 7.6. An example graph with maximum degree 2.

(a sequence with such and such properties) or a graph (normally, a subgraph of some other host graph); the intended meaning shall be clear from context.

7.5 Exercise. Let $G := (V, E, \varphi)$ be a graph. Prove[i] that, if $\Delta(G) \leq 2$, then every component of G is a path or a cycle; see Figure 7.6.

7.6 Exercise. Let $G := (V, E, \varphi)$ be a graph. Suppose that M and N are matchings in G. Set

$$H_\cup := (V, M \cup N, \varphi\restriction_{M \cup N}),$$

[i] *Hint:* Start with a longest (maximum-length) path in G and consider the two possibilities for the degree of its endpoints. Alternatively, to make the proof more algorithmic, find a maximal subgraph that is a path. The notion of maximality is defined in the assignments.

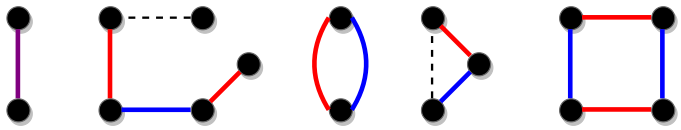


FIGURE 7.7. Two matchings (one red and one blue) in a graph; the dashed edges are in the graph but in neither of these two matchings, and the purple edge is in both matchings.

$$H_\Delta \coloneqq (V, M \,\Delta\, N, \varphi\restriction_{M\Delta N}).$$

Prove that, if C is a component of $H_\cup$ or H_Δ, then exactly one of the following holds:

(7.5)
- (i) C has precisely two vertices joined by an edge in $M \cap N$, and this only occurs for $H_\cup$;
- (ii) C is an even cycle that is both M- and N-alternating;
- (iii) C is an even path that is both M- and N-alternating;
- (iv) C is an odd path that is both M- and N-alternating.

In (7.5)(i) to (7.5)(iii), the component C has the same number of edges in M and in N, whereas in (7.5)(iv), either one of the matchings M or N has one more edge in C than the other matching. See Figure 7.7.

7.7 Theorem (Berge's theorem). Let $G \coloneqq (V, E, \varphi)$ be a graph, and let M be a matching in G. Then M is a maximum matching if and only if there is no M-augmenting path.

Proof. Necessity: the contrapositive of necessity appears in §7.3.

Sufficiency: We prove the contrapositive. Suppose N is a matching in G such that $|N| > |M|$. By Exercise 7.6, the set $\mathcal{C}$ of components of $(V, M \cup N, \varphi\restriction_{M\cup N}) =: H_\cup$ is made of paths and cycles that are M- and N-alternating.

(7.6) Suppose that no element of $\mathcal{C}$ is an M-augmenting path.

Then, by (7.5), every component C of $H_\cup$ has at least as many edges in M than in N:

$$|E(C) \cap M| \geq |E(C) \cap N|. \qquad \text{for each } C \in \mathcal{C}. \tag{7.7}$$

We have

$$\begin{aligned} M &= E(H_\cup) \cap M && \text{since } M \subseteq E(H_\cup) \\ &= \textstyle\bigcup_{C\in\mathcal{C}} \big(E(C) \cap M\big) && \text{since } E(H_\cup) = \textstyle\bigcup_{C\in\mathcal{C}} E(C). \end{aligned} \tag{7.8}$$

and similarly

$$N = \textstyle\bigcup_{C\in\mathcal{C}} \big(E(C) \cap N\big) \tag{7.9}$$

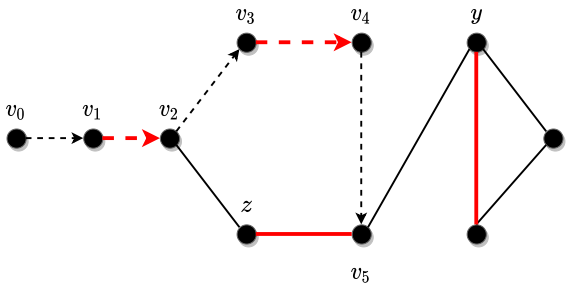
Using that the sets in $\{\, E(C) : C \in \mathcal{C}\}$ are pairwise disjoint, we get

$$\begin{aligned} |M| &= \textstyle\sum_{C\in\mathcal{C}} \big|E(C) \cap M\big| && \text{by (7.8)} \\ &\geq \textstyle\sum_{C\in\mathcal{C}} \big|E(C) \cap N\big| && \text{by (7.7)} \\ &= |N| && \text{by (7.9).} \end{aligned}$$

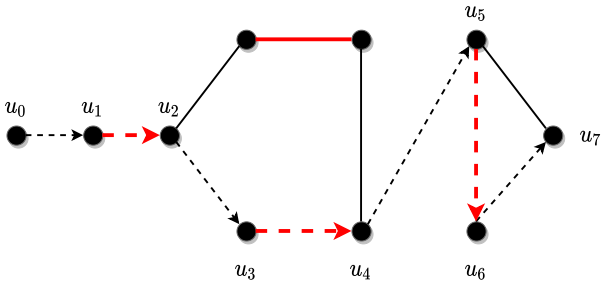
This is a contradiction, since $|N| > |M|$. Hence, the assumption (7.6) is false, and the result is proved. □

7.8 Towards an Algorithm, and Certificates. Berge's Theorem 7.7 shows that, in order to find a maximum matching in an input graph, it suffices to iterate the following procedure, starting with the empty matching $M_0 \coloneqq \varnothing$: find an M_t-augmenting path P_t for the current matching M_t, then set $M_{t+1} \coloneqq M_t \,\Delta\, E(P_t)$. This statement is correct, but it is tricky in that, for the "suffices" part, it is highly nontrivial to design an efficient algorithm that finds augmenting paths if they exist. See Figure 7.8. In fact, it took the genius of Jack Edmonds [4] to design such an algorithm, in a landmark paper that also introduced our convention of calling an algorithm efficient if it runs in polynomial time.

While Theorem 7.7 does suggest an algorithm, it does not provide a certificate, in the sense discussed in §3.15. Indeed, in the case where an algorithm just concludes that there is augmenting path (and nothing else), the best way to convince the caller is to say: "here, I conducted my search and found nothing, if you want I can walk you through the whole search again", to which the caller should probably just say "thanks,



(A) When searching for an M-augmenting path starting at v_0, upon reaching v_2, one has the option of proceeding upwards to v_3 or downwards to z. Then, as one reaches v_5, the edge that goes right to y is not accessible for extending the M-alternating walk that is being grown. At this point, the M-alternating walk can only be grown by proceeding to z, and then one gets stuck.



(B) As in Figure 7.8a, if one reaches u_2 but then proceeds downwards, an M-augmenting path can be found.

FIGURE 7.8. These examples illustrate why it is nontrivial to search for an M-augmenting path. Namely, there are options to be taken at some steps, and it is not clear how to avoid backtracking.

but no thanks". The issue here is that Berge's Theorem 7.7 provides no *object* to return to the caller so that she can be convinced.

7.9 Bipartite Graphs. Let $G \coloneqq (V, E, \varphi)$ be a graph. We call G **bipartite** if there are disjoint subsets $U, W \subseteq V$ such that every edge $e \in E$ satisfies $\varphi(e) \cap U \neq \varnothing \neq \varphi(e) \cap W$, i.e., each edge has an end in U and the other end in W. If so, we can say that G is (U, W)**-bipartite**; note, however, that, if G is bipartite, the sets U and W above are not uniquely determined. In the context above, the sets U and W are usually referred to as **color classes** of G. Figure 7.1b shows an example of a bipartite graph.

7.10 Exercise. Let $G \coloneqq (V, E, \varphi)$ be a graph. Prove that precisely one of the following holds:

(7.10) (i) G is bipartite, or
(ii) G has an odd cycle.

The proof can be made algorithmic, by adapting BFS or DFS, so determining which option from (7.10) holds, and providing the corresponding certificate (color classes in case of (7.10)(i), or an odd cycle in case of (7.10)(ii)), can be done in linear time.

7.11 Problem (Bipartite Matching). We will focus on the Maximum Matching Problem 7.2 only for bipartite graphs, at least for now. The problem is: **given** a bipartite graph $G \coloneqq (V, E, \varphi)$, and sets $U, W \subseteq V$ such that G is (U, W)-bipartite, solve the optimization problem (7.1).

7.12 Motivation. We illustrate a motivation for solving Problem 7.11 (and thus also[i] Problem 7.2) by an example problem. Suppose we have a set U of agents and a set W of tasks to be performed. Add an edge between an

[i]Displaying a quick motivation for the more general Problem 7.2 is a bit harder, and we refrain from doing so here. Let it suffice to mention that it is an extremely important and powerful model, which can be used to attack other, obviously-applicable problems.

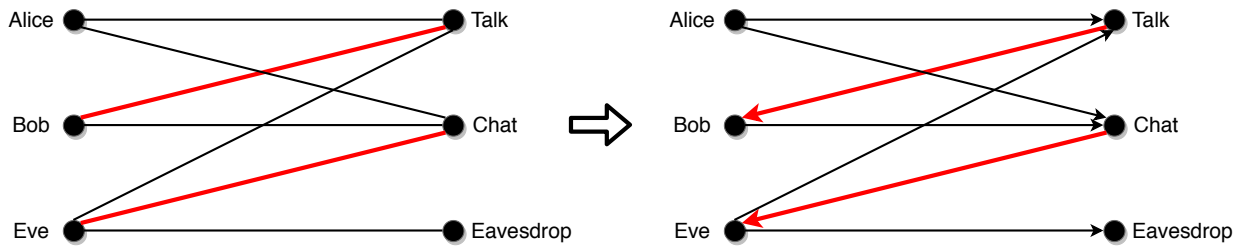


FIGURE 7.9. A matching M in a bipartite graph G, along with the orientation from Exercise 7.14.

agent and a task if said agent is qualified/trained to perform said task. The agents cannot multitask: they can only perform one task at any given time. Then the maximum matching problem determines an allocation of the maximum number of tasks that can be performed simultaneously.

7.13 Orientations. A digraph $D := (V, A, \psi)$ is an **orientation** of a graph $G := (V, E, \varphi)$ if $A = E$ and, for each $a \in A$, if $\psi(a) = (u, v)$ for some $u, v \in V$, then $\varphi(a) = \{u, v\}$. In this case, we say that G is the **underlying graph** of D.

For instance, if $G := (V, E, \varphi)$ is a (U, W)-bipartite graph, we might create an orientation $D := (V, E, \psi)$ of G by setting, for each edge $e \in E$, the arc $\psi(e) := (u, w)$, where u and w are the ends of e in U and W, respectively.

7.14 Exercise (Augmenting Paths in Bipartite Graphs). Let $G := (V, E, \varphi)$ be a (U, W)-bipartite graph, and let $M \subseteq E$ be a matching in G. Define the orientation $D(G, U, W, M) := (V, E, \psi)$ of G as follows: for each edge $e \in E$, let u and w be the ends of e in U and W, respectively, and set

$$\psi(e) := \begin{cases} (u, w) & \text{if } e \notin M, \\ (w, u) & \text{if } e \in M. \end{cases} \tag{7.11}$$

See Figure 7.9. Let $P := \langle v_0, \ldots, e_\ell, v_\ell \rangle$ be a path in G from a vertex v_0 in $U \setminus V_M$ to a vertex v_ℓ either in U or in $W \setminus V_M$. Prove that P is M-alternating if and only if P is a path in $D(G, U, W, M)$. In this case, if $v_\ell \in U$, then

(7.12) (i) either $\ell = 0$,
(ii) or $\ell \geq 1$ and $e_\ell \in M$.

7.15 In the context of Exercise 7.14, every M-augmenting path $Q := \langle v_0, e_1, v_1, \ldots, e_\ell, v_\ell \rangle$ is odd, so either Q starts at some vertex in $U \setminus V_M$ and ends at some vertex in $W \setminus V_M$, or else the "reverse" $\langle v_\ell, e_\ell, v_{\ell-1}, \ldots, e_1, v_0 \rangle$ of Q has this form. That is, in searching for M-augmenting paths, it suffices to search for M-augmenting paths from $U \setminus V_M$ to $W \setminus V_M$. And by Exercise 7.14, that reduces to finding ordinary paths in the digraph $D(G, U, W, M)$, which can be done using BFS or DFS. So we have all the ingredients for an efficient algorithm for the Bipartite Matching Problem 7.11, though still no certificate.

LECTURE 8: VERTEX COVERS AND KŐNIG'S ALGORITHM

8.1 Vertex Covers. Let $G := (V, E, \varphi)$ be a graph. A subset $K \subseteq V$ is a **vertex cover** (of G) if every edge of G has an end in K, that is, for each edge $e \in E$, one has $\varphi(e) \cap K \neq \varnothing$. See the example in Figure 8.1.

The **minimum vertex cover problem** is, **given** a graph $G := (V, E, \varphi)$, solve the optimization problem

$$\begin{array}{ll} \text{Minimize} & |K| \\ \text{subject to} & K \text{ is a vertex cover of } G. \end{array} \tag{8.1}$$

Note that (8.1) is feasible for every input graph G, since $V(G)$ is a feasible solution. Thus,

(8.1) has an optimal solution

by Theorem 1.5. The **vertex cover number** of G, denoted by $\tau(G)$, is the optimal value of (8.1).

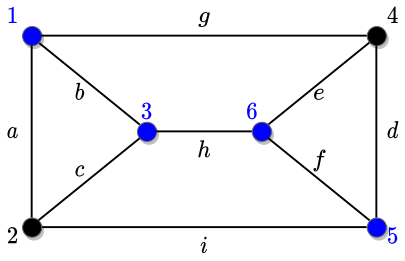


FIGURE 8.1. A vertex cover K of a graph G which shows that $\tau(G) \leq 4$; verify that equality holds, so that the inequality (8.4) may be strict.

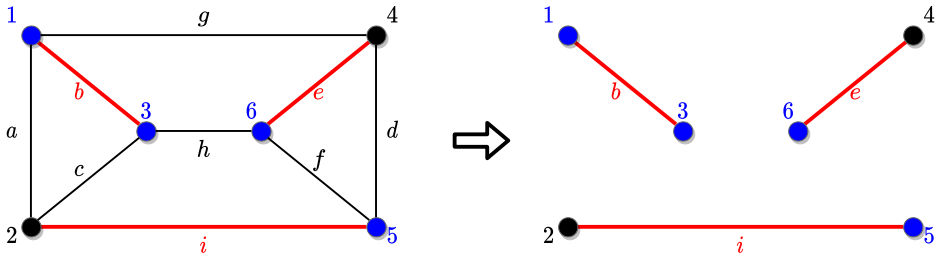


FIGURE 8.2. A matching M and a vertex cover K of a graph $G = (V, E, \varphi)$ on the left, and the spanning subgraph $H = (V, M, \varphi\restriction_M)$ on the right.

8.2 Theorem. Let $G \coloneqq (V, E, \varphi)$ be a graph. Let $M \subseteq E$ be a matching in G, and let $K \subseteq V$ be a vertex cover of G. Then

$$|M| \leq |K|, \tag{8.2}$$

(8.3) with equality in (8.2) if and only if $K \subseteq V_M$ and $|\varphi(e) \cap K| = 1$ for each $e \in M$.

In particular,

$$\nu(G) \leq \tau(G). \tag{8.4}$$

Proof. It is easy to verify that, if H is a spanning subgraph of G, then K is a vertex cover of H as well. By taking $H \coloneqq (V, M, \varphi\restriction_M)$, we find that K is a vertex cover of H. However, since H has $|M|$ pairwise disjoint edges, it is clear that $|K| \geq |M|$; see Figure 8.2. Also, equality holds if and only if each vertex of K is M-saturated and K has precisely one end of each edge of M, thus proving (8.3). Equation (8.4) follows immediately from (8.2). □

8.3 Corollary. Let $G \coloneqq (V, E, \varphi)$ be a graph. Let $M^* \subseteq E$ be a matching in G, let $K^* \subseteq V$ be a vertex cover of G, and suppose that

$$|K^*| = |M^*|. \tag{8.5}$$

Then M^* is a maximum matching, i.e., M^* is an optimal solution for (7.1), and K^* is a minimum vertex cover, i.e., an optimal solution for (8.1).

Proof. Let $M \subseteq E$ be a matching in G, and let $K \subseteq V$ be a vertex cover of G. Then

$$\begin{aligned} |K| &\geq |M^*| && \text{by Theorem 8.2} \\ &= |K^*| && \text{by (8.5).} \end{aligned}$$

Hence, K^* is optimal for (8.1). Similarly,

$$\begin{aligned} |M| &\leq |K^*| && \text{by Theorem 8.2} \\ &= |M^*| && \text{by (8.5).} \end{aligned}$$

Hence, M^* is optimal for (7.1). □

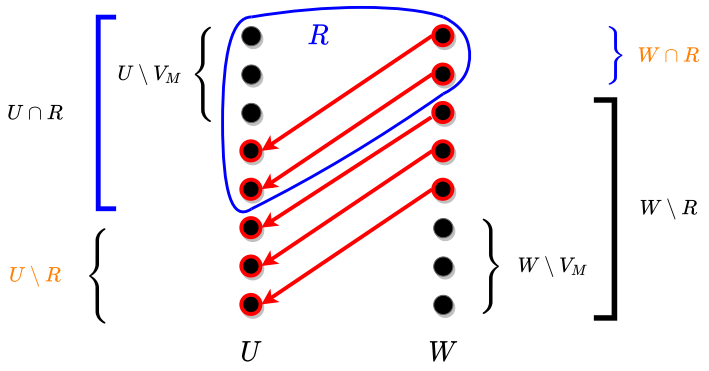


FIGURE 8.3. Schematic illustration for the vertex cover $K = (U \setminus R) \cup (W \cap R) \subseteq V$ from Theorems 8.5 and 8.6, for the (U, W)-bipartite graph $G = (V, E, \varphi)$. The matching edges appear as arcs from the orientation $D \coloneqq D(G, U, W, M)$, and R is the set of vertices reachable in D from $U \setminus V_M$.

8.4 Theorem 8.2 and Corollary 8.3 should remind you very strongly of Theorem 4.2 and Corollary 4.3. Results such as these will recur throughout the course, and they will provide elegant certificates of optimality for both the minimization and maximization problems. The graph in Figure 8.1 shows that the inequality (8.4) can be strict. This is probably related (though we will not explain why) to the fact that the problem (8.1) is NP-hard (see [3, Theorem 34.12]). We next proceed to show that (8.4) holds with equality whenever the graph is bipartite. The next algorithm builds upon the ideas outlined in §7.8 and §7.15, combined with the upper bound from Theorem 8.2.

This is a good point for you to peruse Kőnig's Algorithm 8.1. In what follows, we prove the correctness and consequences of Algorithm 8.1. Note also that Algorithm 8.1 runs in polynomial time, since there are $O(n)$ iterations of the **for** in Line 4, each one of which can be implemented to run in time $O(n+m)$; see §3.9.

Algorithm 8.1 Kőnig's Algorithm for Problem 7.11

Input: a tuple (G, U, W), formed by
(i) a bipartite graph $G = (V, E, \varphi)$,
(ii) subsets $U, W \subseteq V$ such that G is (U, W)-bipartite
Output: a matching $M^* \subseteq E$ in and a vertex cover $K^* \subseteq V$ such that $|M^*| = |K^*|$, as in Corollary 8.3.

```
1:  function KŐNIG(G, U, W)
2:      M_0 ← ∅
3:      n ← |V|
4:      for t ← 0 to n/2 do
5:          D_t ← D(G, U, W, M_t), defined as in Exercise 7.14
6:          R_t ← { v ∈ V : there is u ∈ U \ V_{M_t} such that u ⇝_{D_t} v }
7:          if R_t ∩ (W \ V_{M_t}) ≠ ∅ then
8:              Let w_t ∈ R_t ∩ (W \ V_{M_t})
9:              Let u_t ∈ U \ V_{M_t} and let P_t be a (u_t, w_t)-path in D_t
10:             M_{t+1} ← M_t Δ E(P_t)
11:         else
12:             return M_t and (U \ R_t) ∪ (W ∩ R_t)
```

8.5 Theorem. Let $G \coloneqq (V, E, \varphi)$ be a (U, W)-bipartite graph, and let $M \subseteq E$ be a matching in G. Define the digraph $D \coloneqq D(G, U, W, M)$ as in Exercise 7.14. Set $R \coloneqq \{ v \in V :$ there is $u \in U \setminus V_M$ such that $u \rightsquigarrow_D v \}$. Then $K \coloneqq \big(U \setminus R\big) \cup \big(W \cap R\big)$ is a vertex cover of G.

See schematic illustration in Figure 8.3.

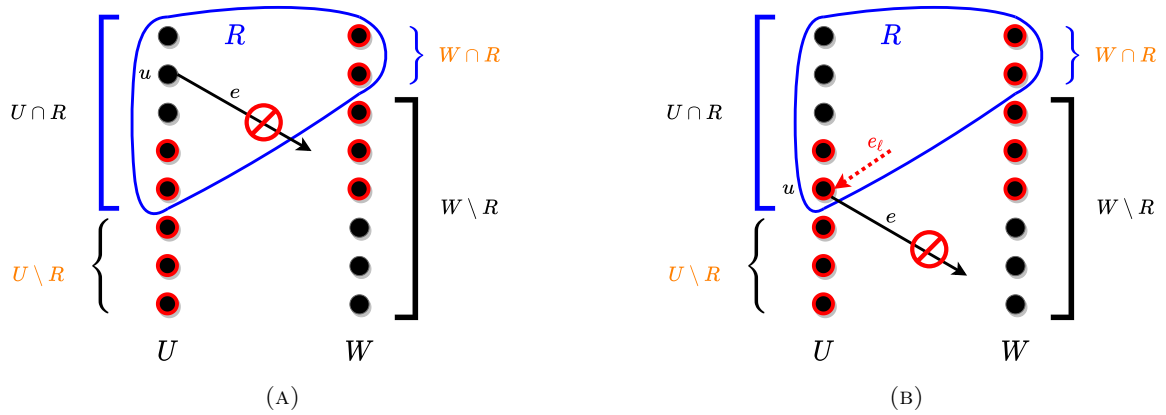


FIGURE 8.4. The two possibilities (8.11) for the "forbidden" edge e in the proof of Theorem 8.5.

Proof. Let $e \in E$ be an edge, and let u and w be the ends of e in U and W, respectively, i.e., $\{u\} = \varphi(e) \cap U$ and $\{w\} = \varphi(e) \cap W$. We want to prove that at least one end of e lies in K.

(8.6) Suppose, for the sake of contradiction, that $u \notin U \setminus R$ and $w \notin W \cap R$.

Since $u \in U$ and $w \in W$, we obtain from (8.6) that

(8.7) $u \in U \cap R$, since $u \in U$ and $U \setminus (U \setminus R) = U \cap R$,

and

(8.8) $w \in W \setminus R$, since $w \in W$ and $W \setminus (W \cap R) = W \setminus R$.

Since $u \in U \cap R$ by (8.7), Corollary 3.2 and Exercise 7.14 show that there is an M-alternating path $P \coloneqq \langle v_0, e_1, v_1, \ldots, e_\ell, v_\ell \rangle$ in G with

(8.9) $v_0 \in U \setminus V_M$,

(8.10) $u = v_\ell$,

and

(8.11) (i) either $\ell = 0$,
(ii) or $\ell \geq 1$ and $e_\ell \in M$.

We will prove that

(8.12) $e \notin M$.

In case (8.11)(i), we have $u = v_0 \in U \setminus V_M$ by (8.10) and (8.9), so $u \notin V_M$ shows that (8.12) holds. Suppose that (8.11)(ii) holds. Since $v_{\ell-1} \in W \cap R$, we have from (8.8) that $v_{\ell-1} \neq w$. In particular, $e \neq e_\ell$. Since $u = v_\ell$ (by (8.10)) is incident with both e and $e_\ell \in M$, it follows[i] that $e \notin M$. This completes the proof of (8.12) in both cases from (8.11). See Figure 8.4.

From (8.12), we find that $\langle u, e, w \rangle$ is a path in D. Hence, the walk $P \cdot \langle u, e, w \rangle$ shows that[ii] $w \in R$. This contradicts (8.8). This contradiction arose from (8.6), which therefore must be false. Falsity of (8.6) means that at least one of the following holds: $u \in U \setminus R$ or $w \in W \cap R$. That is, at least one end of e lies in K, whence K is a vertex cover, and the proof is complete. □

8.6 Theorem. Let $G \coloneqq (V, E, \varphi)$ be a (U, W)-bipartite graph, and let $M \subseteq E$ be a matching in G. Define the digraph $D \coloneqq D(G, U, W, M)$ as in Exercise 7.14. Set $R \coloneqq \{ v \in V :$ there is $u \in U \setminus V_M$ such that $u \rightsquigarrow_D v \}$. If

(8.13) $R \cap (W \setminus V_M) = \varnothing$,

then $K \coloneqq \big(U \setminus R\big) \cup (W \cap R)$ is a vertex cover of G and $|K| = |M|$.

[i] Since M is a matching.
[ii] Alternatively, note that $e \in \delta^{\text{out}}_D(R)$ by (8.12), however the latter set is empty by Exercise 3.12.

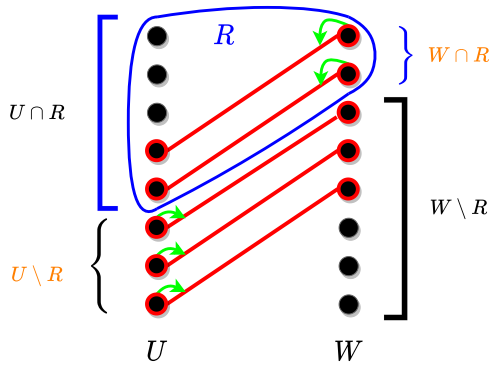


FIGURE 8.5. A function f from $K = (U \setminus R) \cup (W \cap R) \subseteq V$ to the matching $M \subseteq E$.

Proof. Theorem 8.5 shows that K is a vertex cover. Hence, $|K| \geq |M|$ by Theorem 8.2. Thus, it suffices to prove the reverse inequality $|K| \leq |M|$. Hence,

(8.14) it suffices to prove that there is an injective function from K to M.

By definition the definition of R, we have

$$U \setminus V_M \subseteq R. \tag{8.15}$$

By taking the complement with respect to U in both sides of (8.15), the inclusion is reversed, and we obtain

$$U \setminus (U \setminus V_M) \supseteq U \setminus R,$$

i.e.,

$$U \setminus R \subseteq U \cap V_M. \tag{8.16}$$

From hypothesis (8.13) and $W \cap R \subseteq W$ we obtain

$$W \cap R \subseteq W \setminus (W \setminus V_M) = W \cap V_M. \tag{8.17}$$

Hence, (8.16) and (8.17) show that $K \subseteq V_M$. In particular, every vertex in K is M-saturated, so there is a function $f \colon K \to M$ such that,

(8.18) for every $v \in K$, the vertex v is incident with the edge $f(v)$.

See Figure 8.5.

We claim that

(8.19) f is injective.

Let $v, z \in K$ such that $f(v) = f(z)$. To prove (8.19), we want to show that $v = z$. By (8.18), vertices v and z are both incident with the edge $f(v) = f(z)$. Since every edge has one end in U and the other in W, we find that, if $v, z \in U$, then v and z are the same vertex, namely, the only end of $f(v)$ in U. Similarly, if $v, z \in W$, then $v = z$. In either case, we are done, so

(8.20) suppose, for the sake of contradiction, that $u := v \in U$ and $w := z \in W$.

Since $u = v \in U$ and $v \in K$, we find that

$$u \in U \cap K = U \setminus R. \tag{8.21}$$

Since $w \in K \subseteq W \cap R$, there is a path P from a vertex in $U \setminus V_M$ to w in D. Set $e := f(u) = f(v) = f(z) = f(w) \in M$. Then $\langle w, e, u \rangle$ is a path in D. Hence, $P \cdot \langle w, e, u \rangle$ is a walk in D, which shows[i] that

$$u \in R. \tag{8.22}$$

This contradicts (8.21). Hence, the assumption in (8.20) is false. This completes the proof of (8.19), and the result is proved by (8.14). □

[i] Alternatively, note that $e \in \delta_D^{\text{out}}(R)$ by (8.12), however the latter set is empty by Exercise 3.12.

8.7 Theorem (Correctness of Kőnig's Algorithm 8.1). Let $G \coloneqq (V, E, \varphi)$ be a (U, W)-bipartite graph. Then the sets $M^* \subseteq E$ and $K^* \subseteq V$ returned by Algorithm 8.1 satisfy the following properties:

(i) M^* is a maximum matching,
(ii) K^* is a vertex cover with $|K^*| = \tau(G)$,
(iii) $|M^*| = |K^*|$.

Proof. It is easy to verify, using Exercise 7.14 and (7.4), that

$$M_t \text{ is a matching of size } |M_t| = t \text{ whenever } M_t \text{ is defined.} \tag{8.23}$$

The vertex set K^* returned in Line 12 is a vertex cover by Theorem 8.5. Moreover, in order to get to Line 12, the condition in Line 7 must have tested negative, so that $|K^*| = |M^*|$ follows from Theorem 8.6. In that case, optimality of both M^* and K^* follow from Corollary 8.3.

Hence, to prove the result, it now suffices to prove that the algorithm reaches Line 12 eventually. Set $t \coloneqq \nu(G) \leq n/2$. Then M_t is a maximum matching by (8.23). When the **for** loop that starts in Line 4 with this specific value of t, the condition in the **if** from Line 7 must be false; otherwise, M_{t+1} will be defined in Line 10 which will then be, by (8.23), a matching of size $t + 1 > t = \nu(G)$, a contradiction. □

8.8 Corollary (Kőnig's matching theorem). Let $G \coloneqq (V, E, \varphi)$ be a bipartite graph. Then the maximum size of a matching in G is equal to the minimum size of a vertex cover of G.

Proof. Immediate from Theorem 8.7. □

Lecture 9: Linear Programming Relaxations

9.1 Linear Programs. Our discussion about linear optimization will remain mostly informal. Recall that, for vectors $y, z \in \mathbb{R}^n$, the so-called dot product of y and z is

$$y^\mathsf{T} z = \sum_{i=1}^{n} y_i z_i; \tag{9.1}$$

Here, $\cdot^\mathsf{T}$ denotes taking the transpose of a vector.

One may think of a linear program[i] (LP) as an optimization problem of the form[1]

$$\text{Maximize} \quad c^\mathsf{T} x \tag{9.2a}$$
$$\text{subject to} \quad a_i^\mathsf{T} x \leq b_i \qquad \text{for each } i \in I_{\leq}, \tag{9.2b}$$
$$a_i^\mathsf{T} x = b_i \qquad \text{for each } i \in I_{=}, \tag{9.2c}$$
$$a_i^\mathsf{T} x \geq b_i \qquad \text{for each } i \in I_{\geq}, \tag{9.2d}$$
$$x \in \mathbb{R}^n, \tag{9.2e}$$

where $c \in \mathbb{R}^n$ is a vector, the sets $I_{\leq}$, $I_{=}$, and $I_{\geq}$ are pairwise disjoint **finite** index[ii] sets, and $a_i \in \mathbb{R}^n$ is a vector and $b_i \in \mathbb{R}$ is a number for each index $i \in I_{\leq} \cup I_{=} \cup I_{\geq}$. The symbol x is used as the "variable" of the LP (9.2). The feasible solutions of (9.2) are the vectors x in $\mathbb{R}^n$ that satisfy the *(linear) inequality* constraints (9.2b) and (9.2d) and the *(linear) equality* constraints (9.2c). Note that the objective function (9.2a) is a linear function[iii] of the variable x.

9.2 Integer Programs. When one starts with a linear program such as (9.2) and adds[iv] a constraint of the form "$x \in \mathbb{Z}^n$", one obtains an integer program[v] (IP). These constraints are sometimes called *integrality constraints*, and vectors all of whose coordinates are integers are called **integral**.

[1] LPs can be minimization problems as well.

[i] There is a trend to eschew the term "program" in favor of "optimization problem", so that "linear program" becomes "linear optimization problem", and so on. This is mostly meant to avoid confusion with computer programming.
[ii] Note that the term "index" has no mathematical meaning.
[iii] Recall that a function $f\colon X \to Y$ between real vector spaces X and Y is linear if, for each $x_1, x_2 \in X$ and $\alpha_1, \alpha_2 \in \mathbb{R}$, one has $f(\alpha_1 x_1 + \alpha_2 x_2) = \alpha_1 f(x_1) + \alpha_2 f(x_2)$.
[iv] Thus, this essentially replaces the constraint (9.2e).
[v] Integer programs are sometimes called "integer linear programs". This is understandable, and it makes sense, since it is also useful to study integer programs which at not linear. However, there is a language drawback, since an "integer linear program" is not itself a linear program.

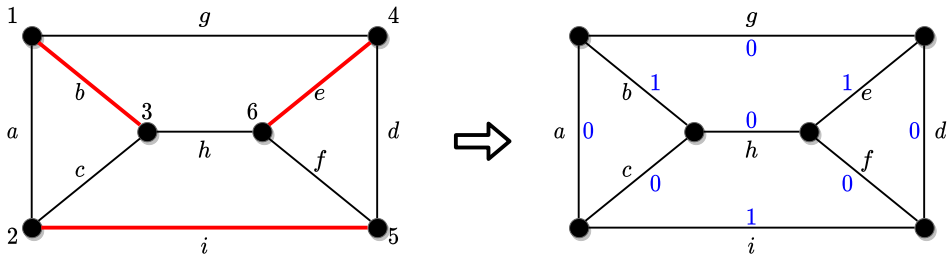


FIGURE 9.1. A matching $M \subseteq E$ on the left, along with a representation of the corresponding $\{0,1\}$-valued vector x^M as in (9.5) on the right, feasible in the LP relaxation of the IP (9.3).

Here is an example of an IP:

$$\begin{aligned}
&\text{(9.3a)} \quad \text{Maximize} && x_a + x_b + x_c + x_d + x_e + x_f + x_g + x_h + x_i && \\
&\text{(9.3b)} \quad \text{subject to} && x_a + x_b \qquad\qquad + x_g \qquad\quad \leq 1, && \\
&\text{(9.3c)} && x_a \qquad + x_c \qquad\qquad\qquad + x_i \leq 1, && \\
&\text{(9.3d)} && x_b + x_c \qquad\qquad\qquad + x_h \qquad \leq 1, && \\
&\text{(9.3e)} && x_d + x_e \qquad + x_g \qquad\quad \leq 1, && \\
&\text{(9.3f)} && x_d \qquad + x_f \qquad\quad + x_i \leq 1, && \\
&\text{(9.3g)} && x_e + x_f \qquad + x_h \qquad \leq 1, && \\
&\text{(9.3h)} && x_k \geq 0 && \text{for each } k \in \{a, b, \ldots, i\}, \\
&\text{(9.3i)} && x_k \in \mathbb{Z} && \text{for each } k \in \{a, b, \ldots, i\}.
\end{aligned}$$

Here, (9.3i) encodes the integrality constraints.

Suppose that x is feasible for (9.3). Then

$$\begin{aligned}
x_a &\leq 1 - x_b - x_g && \text{by (9.3b)} \\
&\leq 1 && \text{by (9.3h)}.
\end{aligned}$$

Similarly, one can show that $x_k \leq 1$ for each $k \in \{b, \ldots, i\}$. Hence, $x_k \in [0,1]$ for each $k \in \{a, \ldots, i\}$, and then the integrality constraint (9.3i) shows that,

(9.4) if x is feasible in the IP (9.3), then x_k is either 0 or 1 for each $k \in \{a, b, \ldots, i\}$.

Consider the maximum matching problem in the graph $G = (V, E, \varphi)$ from Figure 7.1a. Note that each of the constraints (9.3b) to (9.3g) corresponds to some vertex v, and the variables with positive coefficients (all equal to 1) in one of these constraints correspond to all the edges with which v is incident. Combined with (9.4), this shows that, in a feasible solution, one may set to 1 at most one variable corresponding to an edge with which each vertex is incident.

Let us formalize this discussion. For each subset $F \subseteq E$, let x^F be the vector in $\mathbb{R}^E$ where

$$x^F(e) := [e \in F] \qquad \text{for each } e \in E. \tag{9.5}$$

Verify that, if $M \subseteq E$ is a matching, i.e., M is feasible in the maximum matching problem (7.1) for our graph G, then x^M is feasible in the IP (9.3) with objective value $|M|$; see Figure 9.1. Similarly, verify that, if $x \in \mathbb{R}^E$ is feasible in the IP (9.3), then $x = x^M$ for some matching M of G, and $|M|$ is the objective value of x in (9.3).

This discussion should convince you that the maximum matching problem (7.1) for this graph G and the integer program (9.3) are isomorphic, with an isomorphism provided by the map described in (9.5).

9.3 Linear Programming Relaxations. The isomorphism described in §9.2 shows that integer programs are useful modeling tools for combinatorial optimization problems. It is not hard to formulate several optimization problems defined over graphs or digraphs as integer programs, using a small collection of modeling "tricks".

Once one formulates a combinatorial optimization problem as an IP, such as (9.3), it is often **extremely useful** to study the *LP relaxation* of the IP, that is, the LP obtained from the IP by *relaxing* the integrality

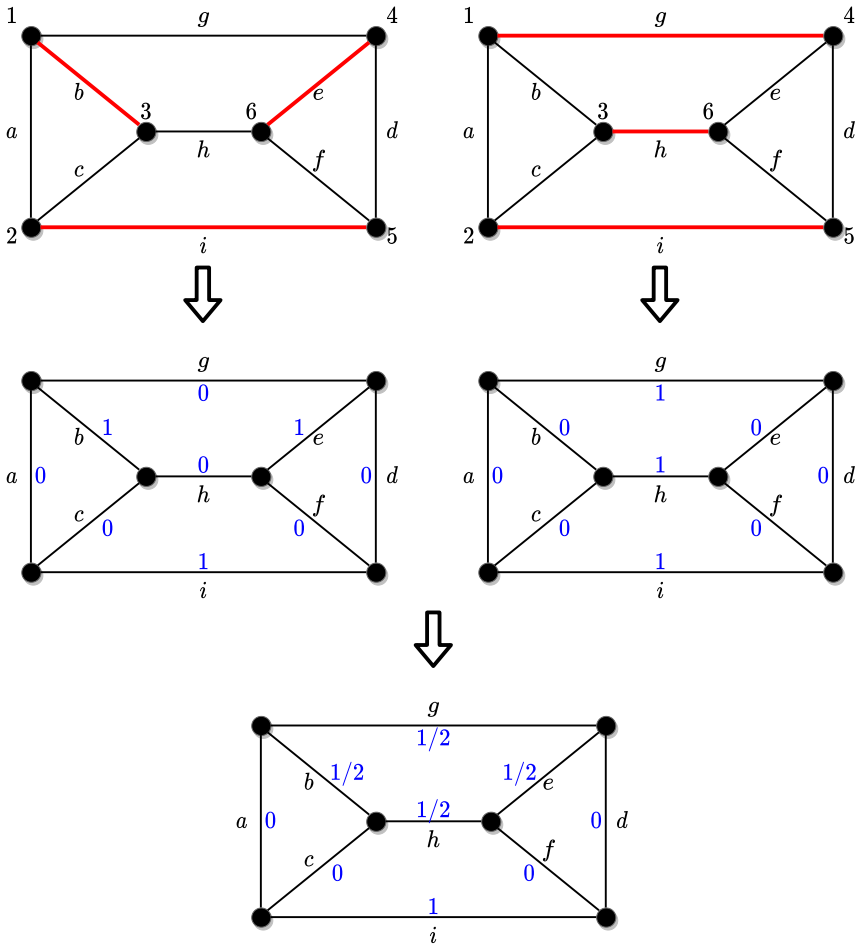


FIGURE 9.2. The two matchings M and N at the top generate $\{0,1\}$-valued vectors vectors x^M and x^N in the middle, which are then averaged to yield a feasible solution $\frac{1}{2}x^M + \frac{1}{2}x^N$ in the LP relaxation of the IP (9.3).

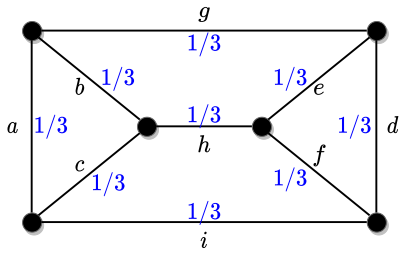


FIGURE 9.3. A vector feasible in the LP relaxation of the IP (9.3) but not arising from matchings of the graph as in Figure 9.2.

constraint "$x \in \mathbb{Z}^n$" to "$x \in \mathbb{R}^n$". In the case of (9.3), one would drop the integrality constraint (9.3i) and study the LP thus obtained.

Figures 9.2 and 9.3 illustrate[i] some feasible solutions for the LP relaxation of (9.3).

[i] It is very hard (impossible?) to "visualize" high-dimensional vectors. However, for (most of) the purposes of the IP (9.3) and its LP relaxation, one need only verify the feasibility of such vectors. Drawing the vector alongside the edges as in Figures 9.1, 9.2, and 9.3 hopefully helps you with this "visualization" task.

Note that the optimal value of LP relaxation of (9.3) is greater than or equal to the optimal value of (9.3), which is the matching number $\nu(G)$ of the graph G from Figure 7.1a; indeed, note that the identity function is a homomorphism from the IP to the LP relaxation, and recall (6.5). Thus, one (preliminary) reason to see why it is useful to study the LP relaxation is that we obtain *upper bounds* for the maximization problem at hand. Recall from §4.4 that upper bounds are the "nontrivial" ones for maximization problems.

9.4 Cuts. Let $G \coloneqq (V, E, \varphi)$ be a graph. For a subset $S \subseteq V$, denote

$$\delta_G(S) \coloneqq \{\, e \in E : \varphi(e) \cap S \neq \varnothing \neq \varphi(e) \setminus S \}$$

be the set of edges with one end in S and another end outside of S. As usual, the index G may be omitted from $\delta_G(S)$ if the graph is clear from context. Note that $\delta(\varnothing) = \delta(V) = \varnothing$. A subset $F \subseteq E$ of edges is called a **cut** if $F = \delta(S)$ for some $S \subseteq V$, and F is a **nontrivial cut** if $F = \delta(S)$ for some $S \subseteq V$ such that $\varnothing \neq S \neq V$. If $v \in V$, we abuse notation and write $\delta_G(v) \coloneqq \delta_G(\{v\})$. We call $\delta(S)$ the **cut induced by** S.

Verify that, if $G \coloneqq (V, E, \varphi)$ is a loopless graph, then

$$\deg_G(v) = |\delta_G(v)| \qquad \text{for each } v \in V. \tag{9.6}$$

9.5 Degree Constraints. Let $G \coloneqq (V, E, \varphi)$ be the graph shown in Figure 7.1a. Verify that each of the inequalities (9.3b) to (9.3g) from the IP (9.3) has the form

$$\sum_{e \in \delta(v)} x_e \leq 1 \tag{9.7}$$

for some $v \in V$. Such constraints are sometimes called *degree constraints*[i].

For a subset $F \subseteq E$ and a vector $x \in \mathbb{R}^E$, denote

$$x(F) \coloneqq \sum_{f \in F} x_f. \tag{9.8}$$

Then the degree constraints of the form (9.7) may be written as $x\big(\delta(v)\big) \leq 1$ for each vertex $v \in V$. There is nothing special about the particular graph G from Figure 7.1a. Indeed, verify that, if $G = (V, E, \varphi)$ is *any* loopless[ii] graph, then once one adds integrality constraints to the LP

$$\begin{aligned}
&\text{Maximize} && x(E) && && \text{(9.9a)}\\
&\text{subject to} && x\big(\delta(v)\big) \leq 1 && \text{for each } v \in V, && \text{(9.9b)}\\
& && x_e \geq 0 && \text{for each } e \in E, && \text{(9.9c)}
\end{aligned}$$

the resulting IP is isomorphic to the maximum matching problem (7.1) over G.

9.6 Vector Ordering. The set

$$\mathbb{R}^n_+ \coloneqq \mathbb{R}^{[n]}_+ = \big\{\, x \in \mathbb{R}^n : x_i \geq 0 \text{ for each } i \in [n] \big\}$$

is called the **nonnegative orthant**[iii]. For vectors $x, y \in \mathbb{R}^n$, write

$$x \geq y \qquad \text{if } x - y \in \mathbb{R}^n_+, \qquad \text{i.e., } x_i \geq y_i \text{ for each } i \in [n]. \tag{9.10}$$

If this is the case, we may also write $y \leq x$. Note that it is **not** the case that, if $x, y \in \mathbb{R}^n$, then $x \leq y$ or $x \geq y$. (You should build an example of this in $\mathbb{R}^2$.)

9.7 Linear Programs with Matrices. Consider the "general form" LP in (9.2). Every linear inequality constraint of the form $a_i^\mathsf{T} x \geq b_i$ can be rewritten as $-a_i^\mathsf{T} x \leq -b_i$. Similarly, every linear equality constraint of the form $a_i^\mathsf{T} x = b_i$ can be written as the two linear inequality constraints $a_i^\mathsf{T} x \leq b_i$ and $a_i^\mathsf{T} x \geq b_i$. By

[i] In connection with (9.6).

[ii] How would you modify the IP so that the same statement would hold for graphs with loops as well?

[iii] If we were considering $\mathbb{R}^2_+$, we would probably call it the nonnegative quadrant. When we consider $\mathbb{R}^3$, the corresponding "quadrant" would be an "orthant", since there are eight of them in $\mathbb{R}^3$. The name for $n = 3$ stuck.

performing this for each index $i \in I_= \cup I_\geq$, we may rename the index sets $I_\leq$, $I_=$, and $I_\geq$ so that both index sets $I_=$ and $I_\geq$ are empty. Then, if $I_\leq = [m]$, one could define the matrix

$$A \coloneqq \begin{bmatrix} — & a_1^\mathsf{T} & — \\ — & a_2^\mathsf{T} & — \\ & \vdots & \\ — & a_m^\mathsf{T} & — \end{bmatrix} \tag{9.11}$$

and *all* the linear inequality constraints in (9.2) can be written compactly in matrix form as '$Ax \leq b$', using the vector ordering $\leq$ from §9.6.

It will be convenient for us to consider LPs where the variables are required to be nonnegative, so we will consider LPs of the form

$$\begin{aligned} &\text{Maximize} && c^\mathsf{T} x && \text{(9.12a)} \\ &\text{subject to} && Ax \leq b, && \text{(9.12b)} \\ & && x \in \mathbb{R}^n_+. && \text{(9.12c)} \end{aligned}$$

Note that we are using the vector ordering $\leq$ from §9.6 in (9.12b).

For example, by setting $E \coloneqq \{a, \ldots, i\}$, the LP relaxation of the IP (9.3) can be written as

$$\begin{aligned} &\text{Maximize} && x_a + \cdots + x_i && \text{(9.13a)} \\ &\text{subject to} && \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{ccccccccc} a & b & c & d & e & f & g & h & i \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix} \end{array} \begin{bmatrix} x_a \\ x_b \\ x_c \\ x_d \\ x_e \\ x_f \\ x_g \\ x_h \\ x_i \end{bmatrix} \leq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, && \text{(9.13b)} \\ & && x \in \mathbb{R}^E_+. && \text{(9.13c)} \end{aligned}$$

9.8 Bounding the Optimal Value with Constraints. Consider again the LP relaxation of the IP (9.3), obtained by dropping the integrality constraint (9.3i). Sum the linear inequalities (9.3b), (9.3d), (9.3f), and (9.3g) to obtain

$$x_a + 2x_b + x_c + x_d + x_e + 2x_f + x_g + 2x_h + x_i \leq 4. \tag{9.14}$$

This shows that every feasible solution x (since x satisfies all linear inequalities) also satisfies (9.14). Hence, every feasible solution x has objective value

$$\begin{aligned} & x_a + x_b + x_c + x_d + x_e + x_f + x_g + x_h + x_i \\ & \quad \leq x_a + 2x_b + x_c + x_d + x_e + 2x_f + x_g + 2x_h + x_i && \text{by (9.3h)} \\ & \quad \leq 4 && \text{by (9.14).} \end{aligned} \tag{9.15}$$

Thus, we obtained an upper bound on the objective value of *any* (!) feasible solution, i.e., on the optimal value of the LP relaxation.

From the viewpoint of the matrix-based formulation (9.13), the inequality (9.14) was obtained by summing rows 1, 3, 5, and 6 of the matrix in (9.13b) to obtain the inequality

$$\begin{matrix} a & b & c & d & e & f & g & h & i \end{matrix} \qquad \begin{bmatrix} x_a \\ x_b \\ x_c \\ x_d \\ x_e \\ x_f \\ x_g \\ x_h \\ x_i \end{bmatrix}$$

(9.16) $$\begin{bmatrix} 1 & 2 & 1 & 1 & 1 & 2 & 1 & 2 & 1 \end{bmatrix} \leq 4.$$

Since each of the coefficients of the variables in the LHS[i] of (9.16) is greater than or equal to the corresponding coefficient in the objective function (all of which are equal to 1), constraint (9.13c) allows us to obtain an upper bound on the objective value of any feasible solution as in (9.15).

9.9 The General Bounding Strategy. The strategy from §9.8 for obtaining upper bounds for the optimal value of an LP may be generalized as follows. Multiply (both sides of) the ith inequality from (9.3b) to (9.3g) by a *nonnegative*[1] number y_i, and sum the scaled inequalities. In the resulting inequality, if each variable x_k has coefficient greater than or equal to 1 (the coefficient of x_k in the objective function (9.3a)), then by (9.3h) the RHS[ii] of the inequality is an upper bound on the objective value of any feasible solution.

From the viewpoint of the matrix-based formulation (9.13), this may be described as follows. Associate a number $y_i \geq 0$ with the ith row of the matrix A in (9.13b). Naturally, y_i is also associated with the RHS of the corresponding row in (9.13b). Multiply the ith row of A by y_i and sum the scaled rows altogether, to obtain a row of coefficients as in the LHS of (9.16). Now, provided that the row of coefficients is greater than or equal to (in the vector ordering from §9.6) the row of coefficients of the objective function, then the RHS of the inequality thus obtained is an upper bound on the objective value of any feasible solution, as in (9.15).

How is one to choose "good" multipliers y_i's for the rows of the matrix? The goal is to obtain the tightest/lowest possible upper bound. The problem of finding the best upper bounds obtained in this way can be formulated as an optimization problem, and in fact as an LP[iii] (!):

(9.17)
$$\begin{array}{lll} \text{Minimize} & y_1 + \cdots + y_6 & \\ & \begin{matrix} & 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} & \\ \text{subject to} & \begin{matrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \\ i \end{matrix} \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{bmatrix} \geq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, & \text{that is} \\ y \in \mathbb{R}^6_+, & \end{array}$$

$$\begin{array}{ll} \text{Minimize} & y_1 + \cdots + y_6 \\ \text{subject to} & y_1 + y_2 \geq 1, \\ & y_1 + y_2 \geq 1, \\ & y_1 + y_3 \geq 1, \\ & y_2 + y_3 \geq 1, \\ & y_4 + y_5 \geq 1, \\ & y_4 + y_6 \geq 1, \\ & y_5 + y_6 \geq 1, \\ & y_1 + y_4 \geq 1, \\ & y_2 + y_5 \geq 1, \\ & y_3 + y_6 \geq 1, \\ & y \in \mathbb{R}^6_+. \end{array}$$

9.10 The Dual LP and Weak Duality. The LP (9.17) is called the LP *dual* of (9.13). Since now we are dealing with two LPs, namely (9.13) and its dual, (9.17), we refer to the original LP, (9.13), as the *primal* LP.

[1] This is required to preserve the inequality.

[i] LHS stands for "left-hand side".
[ii] RHS stands for "right-hand side".
[iii] If you are feeling a bit rusty with matrix manipulation skills, you may find it hard to follow the details of the construction of the LP in (9.17). This is undesirable, but okay. Our goal here is only to connect the dots. Later we will recall matrix manipulations and you should return to this point to make sure you understand. For now, it suffices to get convinced that what is being said is plausible.

By construction, if x is feasible in the primal and y is feasible in the dual, then the objective value of x is less than or equal to the objective value[i] of y.

The definition of the dual LP extends[ii] to any LP in the form (9.12). Namely, the dual LP of (9.12) is

$$\text{Minimize} \quad b^{\mathsf{T}} y \tag{9.18a}$$

$$\text{subject to} \quad y \in \mathbb{R}^m_+, \tag{9.18b}$$

$$A^{\mathsf{T}} y \geq c. \tag{9.18c}$$

Again, by construction, if x is feasible in the primal and y is feasible in the dual, then the objective value of x is less than or equal to the objective value of y. This relation is known as *weak duality*. It should remind you very strongly of our discussions in §4.4 and §8.4.

9.11 Vertex Covers. Recall that the LP (9.13) is a special case of the LP (9.9) applied to the graph from Figure 7.1a. By applying the same construction above, one can setup the dual LP of the more general LP (9.9) for an arbitrary loopless graph $G \coloneqq (V, E, \varphi)$:

$$\text{Minimize} \quad y(V) \tag{9.19a}$$

$$\text{subject to} \quad y\big(\varphi(e)\big) \geq 1 \qquad \text{for each } e \in E, \tag{9.19b}$$

$$y_v \geq 0 \qquad \text{for each } v \in V. \tag{9.19c}$$

Recall that we are using the notation (9.8) in (9.19b), and verify that (9.19) becomes (9.17) when specialized to the graph from Figure 7.1a.

Verify that, if $K \subseteq V$ is a vertex cover of G, then the $\{0,1\}$-valued vector $y^K \subseteq \mathbb{R}^V$ defined analogously as (9.5) is feasible in the LP (9.19), and its objective value is $y^K(V) = |K|$. The connection between weak duality and our discussions in §4.4 and §8.4 should be screaming at the top of its lungs by now.

9.12 Exercise. Use Corollary 8.8 to show that, if $G \coloneqq (V, E, \varphi)$ is a bipartite graph, then the LP (9.9) and its dual LP (9.19) have optimal solutions that are integral.

9.13 Exercise. Let $G \coloneqq (V, E, \varphi)$ be a graph. Show that the LP (9.19) has a feasible solution with objective value $|V|/2$. Use the graph from Figure 7.1a to conclude that the LP (9.9) may have an integral optimal solution[iii] even when G is not bipartite.

Lecture 10: Incidence Matrices

10.1 Vectors. A **vector** is a real-valued function on a finite set, i.e., a function of the form $x \colon V \to \mathbb{R}$ for some finite set V. If $x \in \mathbb{R}^V$ is a vector and $v \in V$, it is usual to denote $x_v \coloneqq x(v)$, and x_v is called the vth **component/entry** of x. **There is no need** for any notion of ordering among the elements of the index set V. In particular, we can have a vector $x \in \mathbb{R}^V$ even if V is the set of dogs, cats, and airplanes. Also, whenever one thinks of an element $x \in \mathbb{R}^n$ as a vector, this is correct, in that $\mathbb{R}^n \coloneqq \mathbb{R}^{[n]}$ by definition[iv], and the fact that there is a known/conventional ordering on the indices is just a coincidence, *which you should overlook*.

The zero vector is denoted by 0, and the ambient space where it lives can be deduced from context (whatever parses). The same applies to the all-ones vector, denoted by $\mathbb{1}$. We already used this implicit notation before, e.g., in the proof of Corollary 3.2 and in §2.6, and we will use it in many other places throughout this text.

The so-called dot product of vectors $x, y \in \mathbb{R}^V$ is[v]

$$x^{\mathsf{T}} y = \sum_{v \in V} x_v y_v. \tag{10.1}$$

[i] Clearly here the objective value of x is the objective value in the primal LP and the objective value of y is the objective value of the dual LP.

[ii] The definition of dual LP extends, in fact, to *any* LP. We are not providing the full details as there is a separate course MAC0315/5790 Linear Optimization, which is not officially a requirement for this course.

[iii] An integral optimal solution is an optimal solution that is integral, and not necessarily an integral solution that is optimal only among integral solutions.

[iv] Note that, by usual set-theoretic conventions, n is the set of all natural numbers strictly smaller than n, so the definition $\mathbb{R}^n \coloneqq \mathbb{R}^{[n]}$ is technically inconsistent. We keep it for the sake of convention and your familiarity with the notation $\mathbb{R}^n$.

[v] This is consistent with (9.1). We will have more to say about this operation in §10.11.

Verify/recall that

$$x^\mathsf{T} y = y^\mathsf{T} x, \tag{10.2}$$

and that

$$x^\mathsf{T}(\alpha y + \beta z) = \alpha x^\mathsf{T} y + \beta x^\mathsf{T} z, \tag{10.3}$$

whenever $x, y, z \in \mathbb{R}^V$ and $\alpha, \beta \in \mathbb{R}$.

10.2 Lecture-Wide Implicit Notation. Throughout the rest of this lecture, we will use the symbols U, V, and W to denote arbitrary *finite* sets, unless explicitly bound otherwise.

10.3 Canonical Basis Vectors. The **canonical basis vectors** in $\mathbb{R}^V$ are the vectors in the set $\{ e_v : v \in V\} \subseteq \{0,1\}^V \subseteq \mathbb{R}^V$, defined by

$$e_v(u) \coloneqq [v = u] \qquad \text{for each } u, v \in V. \tag{10.4}$$

So each vector $e_v \in \mathbb{R}^V$ has precisely one entry equal to one, and all others equal zero.

10.4 Exercise. Verify that

$$\mathbb{1} = \sum_{v \in V} e_v, \tag{10.5}$$

$$e_u^\mathsf{T} x = x_u \qquad \text{for each } u \in V \text{ and } x \in \mathbb{R}^V, \tag{10.6}$$

$$e_u^\mathsf{T} e_v = [u = v] \qquad \text{for each } u, v \in V, \tag{10.7}$$

$$x = \sum_{v \in V} x_v e_v, \qquad \text{for each } x \in \mathbb{R}^V. \tag{10.8}$$

10.5 Incidence Vectors. The **characteristic vector** or **incidence vector** of $U \subseteq V$ is[i]

$$\mathbb{1}_U \coloneqq \sum_{u \in U} e_u \in \{0,1\}^V. \tag{10.9}$$

Verify that the vector x^F defined in (9.5) is equal to $\mathbb{1}_F \subseteq \mathbb{R}^E$, and that

$$e_v^\mathsf{T} \mathbb{1}_U = [v \in U], \tag{10.10}$$

$$\mathbb{1}_U^\mathsf{T} \mathbb{1}_W = |U \cap W| \tag{10.11}$$

for each $v \in V$ and $U, W \subseteq V$. As in (9.8), we abbreviate[ii]

$$x(U) \coloneqq \mathbb{1}_U^\mathsf{T} x \qquad \text{for each } x \in \mathbb{R}^V. \tag{10.12}$$

As you can probably deduce from our use of incidence vectors in (9.5) and the "combinatorial" properties of incidence vectors in (10.7), (10.10), and (10.11), incidence vectors are the main bridge between combinatorial optimization (in discrete objects, such as subsets of vertices, edges, arcs, and the like) and linear optimization (described using vectors and such).

10.6 Matrices. A **matrix** is a real-valued function on the Cartesian product of two finite sets, i.e., a function of the form $A \colon V \times W \to \mathbb{R}$ for some finite sets V and W. Hence, literally, we have $A \in \mathbb{R}^{V \times W}$, using the notation for sets of functions, and we call A a $V \times W$ **matrix**. If $A \in \mathbb{R}^{V \times W}$ is a matrix and $(v, w) \in V \times W$, it is usual to denote $A_{vw} = A_{(v,w)} \coloneqq A((v,w))$, and A_{vw} is called the vw-th **entry** of A. You may think of V as the set of row indices and W as the set of column indices of A.

---

[i] It is quite common to see the vector $\mathbb{1}_U$ denoted in other texts as χ^U. This has a few drawbacks; e.g., when writing on paper or blackboards/whiteboards, the Greek letter chi is quite similar to the English/Latin letter x. Also, it is quite common to need to refer to the transpose of incidence vectors in more general optimization contexts (e.g., in the course MAC0343/6908 Semidefinite Optimization and Applications), and the double superscript in $(\chi^U)^\mathsf{T}$ is typographically cumbersome. One advantage of $\mathbb{1}_U$ is that $\mathbb{1}_V = \mathbb{1}$.

[ii] This notation '$x(U)$' is quite widespread and convenient, even though it clashes with other, equality widespread math notation. Namely, since $x \in \mathbb{R}^V$ is a function and U is a subset of the domain V of x, then $x(U)$ is the *image* of U under x, i.e., $x(U) = \{ x(u) : u \in U\}$. Hence, the RHS of (10.12) is much better in this respect, since it presents no conflict.

As in §10.1, **there is no need** for any notion of ordering among the elements of sets V or W. For instance, the matrix $A \in \mathbb{R}^{[2]\times[3]}$ defined by $A\colon (i,j) \in [2]\times[3] \mapsto i+j \in \mathbb{R}$ can be represented graphically both as

$$A = \begin{array}{c} \\ 1 \\ 2 \end{array}\!\!\begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 2 & 3 & 4 \\ 3 & 4 & 5 \end{bmatrix} \end{array} \qquad \text{and as} \qquad A = \begin{array}{c} \\ 2 \\ 1 \end{array}\!\!\begin{array}{c} \begin{array}{ccc} 3 & 1 & 2 \end{array} \\ \begin{bmatrix} 5 & 3 & 4 \\ 4 & 2 & 3 \end{bmatrix} \end{array}.$$

When the ordering of the indices is clear from context, it is okay to omit the indices in the graphical representation, and then we could write

$$A = \begin{bmatrix} 2 & 3 & 4 \\ 3 & 4 & 5 \end{bmatrix},$$

now using the conventional ordering of numbers. Keep in mind though: this is only a *graphical representation*, the matrix object itself is agnostic about the ordering.

The zero matrix 0 in $\mathbb{R}^{V\times W}$ is the identically zero function on $V\times W$; see the comment about implicit notation for vectors in §10.1.

10.7 Adjacency and Incidence Matrices of Graphs. Let $G=(V,E,\varphi)$ be a graph. The[1] **adjacency matrix** $A_G \in \mathbb{R}^{V\times V}$ of G is defined by

$$A_G(uv) \coloneqq \sum_{e\in E} \big[\varphi(e) = \{u,v\}\big] \qquad \text{for each } uv \in V\times V \tag{10.13}$$

and the $(V\times E)$ **incidence matrix** $B_G \in \{0,1\}^{V\times E}$ of G is defined by

$$B_G(ve) \coloneqq \big[v \in \varphi(e)\big] \qquad \text{for each } ve \in V\times E. \tag{10.14}$$

The matrices below illustrate the adjacency and incidence matrices of the graph G shown in Figure 7.1a:

$$A_G = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array}\!\!\begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \end{array}, \qquad B_G = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array}\!\!\begin{array}{c} \begin{array}{ccccccccc} a & b & c & d & e & f & g & h & i \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix} \end{array}. \tag{10.15}$$

Inspect that the matrix B_G in (10.15) is the same as the matrix in the LHS of (9.13b).

10.8 Householder convention. We strive to maintain consistency with the following modification of the so-called Householder convention:

(i) scalars (i.e., real numbers) are denoted by lowercase Greek letters, such as α, β, γ, and so on;
(ii) vectors are denoted by lowercase English/Latin letters, such as a, b, c, and so on;
(iii) matrices are denoted by uppercase English/Latin letters, such as A, B, C, and so on.

10.9 Matrix Multiplication. If $A \in \mathbb{R}^{U\times V}$ and $B \in \mathbb{R}^{V\times W}$ are matrices, the **product** of A and B is the matrix $AB \in \mathbb{R}^{U\times W}$ defined by

$$(AB)_{uw} \coloneqq \sum_{v\in V} A_{uv}B_{vw} \qquad \text{for each } (u,w) \in U\times W. \tag{10.16}$$

This product has all the basic properties from matrix multiplication with which you are familiar[i], e.g.,

$$(AB)C = A(BC) =: ABC, \tag{10.17a}$$

$$(\alpha A + \beta B)C = \alpha AC + \beta BC, \tag{10.17b}$$

$$C(\alpha A + \beta B) = \alpha CA + \beta CB, \tag{10.17c}$$

[1] Note that we are calling A_G *the* adjacency matrix of G. This is not possible if matrices are required to be indexed by sets of the form $[n]$ with $n \in \mathbb{N}$, unless one requires (shudder) graphs to have vertex set of the form $[n]$ for some $n \in \mathbb{N}$. Think about how this comment would apply to incidence matrices of graphs.

[i] If you like the graphical representation for matrices and vectors to help computing the product AB, it should be clear that, when you draw A with a certain arbitrary ordering of the columns, the same ordering must be used for the rows of B.

whenever A, B, C are matrices and $\alpha, \beta \in \mathbb{R}$, assuming compatibility of dimensions for computing each of the products in (10.17). Recall that matrix multiplication need *not* be commutative, i.e., $AB = BA$ need **not** hold.

10.10 The Transpose. The **transpose** of a matrix $A \in \mathbb{R}^{U \times V}$ is the matrix $A^\mathsf{T} \in \mathbb{R}^{V \times U}$ defined by

$$A^\mathsf{T}_{vu} \coloneqq (A^\mathsf{T})_{vu} \coloneqq A_{uv} \qquad \text{for each } (u, v) \in U \times V. \tag{10.18}$$

Clearly,

$$A^{\mathsf{TT}} \coloneqq \left(A^\mathsf{T}\right)^\mathsf{T} = A. \tag{10.19}$$

Verify that, if A and B are matrices such that the product AB is defined, then[i]

$$(AB)^\mathsf{T} = B^\mathsf{T} A^\mathsf{T}. \tag{10.20}$$

10.11 Vectors as Columns. We *identify* the set $\mathbb{R}^V$ of vectors with the set $\mathbb{R}^{V \times \{\varnothing\}}$ of matrices, where the only index[ii] of a column is $\varnothing$. In other words, vectors can be represented graphically in the same manner as matrices, and they are "column vectors". In particular, If $A \in \mathbb{R}^{U \times V}$ is a matrix and $x \in \mathbb{R}^V$ is a vector, then for the purposes of parsing[iii] the expression Ax, we consider x to be in $\mathbb{R}^{V \times \{\varnothing\}}$, and this makes $y \coloneqq Ax$ an element of $\mathbb{R}^{U \times \{\varnothing\}}$, from which we recover $y \in \mathbb{R}^U$ through the identification[1,2].

By means of our identification of vectors and "column vectors", we see that, in order to write a vector $x \in \mathbb{R}^V$ as a "row vector", one needs to transpose it: $x^\mathsf{T} \in \mathbb{R}^{\{\varnothing\} \times V}$ has a single row in the graphical representation. This is why (9.11) uses the transposes of the vectors a_i's; see also (9.16).

We also *identify* the set of matrices $\mathbb{R}^{\{\varnothing\} \times \{\varnothing\}}$ with $\mathbb{R}$. By means of this identification, the dot product from (10.1) becomes a special case of matrix multiplication[iv].

10.12 Vector Ordering. The vector ordering from §9.6 is easily transferred to elements of $\mathbb{R}^V$: we call the set

$$\mathbb{R}^V_+ = \left\{ x \in \mathbb{R}^V : x_v \geq 0 \text{ for each } v \in V \right\},$$

the **nonnegative orthant**, and for vectors $x, y \in \mathbb{R}^V$, we write

$$x \geq y \qquad \text{if } x - y \in \mathbb{R}^V_+, \qquad \text{i.e., } x_v \geq y_v \text{ for each } v \in V. \tag{10.21}$$

If this is the case, we may also write $y \leq x$.

10.13 Linear Programs. For the purposes of this text, a **linear program** is an optimization problem of the form

$$\text{Maximize} \quad c^\mathsf{T} x \tag{10.22a}$$

$$\text{subject to} \quad Ax \leq b, \tag{10.22b}$$

$$x \in \mathbb{R}^V, \tag{10.22c}$$

for some matrix $A \in \mathbb{R}^{U \times V}$, and vectors $b \in \mathbb{R}^U$ and $c \in \mathbb{R}^V$, or an optimization problem of the same form with 'Minimize' in place of 'Maximize'.

In particular, (10.22c) may be replaced with '$x \in \mathbb{R}^V_+$' and the resulting optimization problem still is an LP (why?[v]).

---

[1] If this *identification* bothers you, it is important to realize that you have been using these types of identifications elsewhere, perhaps unknowingly. For instance, when one constructs the set $\mathbb{R}$ of real numbers from the set $\mathbb{Q}$ of rational numbers, the elements of $\mathbb{R}$ which are called rational are *not* the same elements of $\mathbb{Q}$, but rather they are a subset of $\mathbb{R}$ whose elements are *identified* with those of $\mathbb{Q}$.

[2] You might wonder why we go to the trouble of making this identification. The reason is that we want to define the product *only once*, between matrices (rather than having a *separate* product for matrix and vector) and then we will use whichever properties the product has for matrices also for products of a matrix and a vector. This is very much aligned with our formal approach in this course, of reusing the API instead of cumbersome and inelegant copy-and-paste arguments.

[i] Verify that (10.2) follows from (10.20) through the identifications in §10.11.

[ii] In a standard construction in set theory, the natural number zero is the empty set $\varnothing$ and the natural number 1 is $\{0\} = \{\varnothing\}$, so the identification works between $\mathbb{R}^V$ and $\mathbb{R}^{V \times 1}$, and the only column index is 0.

[iii] Verify that, if vectors were considered as "row vectors" in $\mathbb{R}^{\{\varnothing\} \times V}$, then the product Ax would not parse. This is the reason why vectors are "column vectors"; the notation Ax is too useful to forgo.

[iv] Verify that (10.2) follows from (10.20) through our identifications.

[v] *Hint*: use the (negatives of the) canonical basis vectors e_v's.

10.14 Column and Row Accessors. Let $A \in \mathbb{R}^{U\times V}$ be a matrix and let $v \in V$. For each $u \in U$, the uth entry of the vector $Ae_v \in \mathbb{R}^U$ is

$$\begin{aligned}(Ae_v)_u &= (Ae_v)_{u\varnothing} && \text{by the identification in §10.11}\\ &= \sum_{w\in V} A_{uw}e_v(w\varnothing) && \text{by the definition (10.16) of the product}\\ &= \sum_{w\in V} A_{uw}e_v(w) && \text{by the identification in §10.11}\\ &= \sum_{w\in V} A_{uw}[v=w] && \text{by (10.4)}\\ &= A_{uv}.\end{aligned} \tag{10.23}$$

In other words,

$$Ae_v \text{ is the } v\text{th column of the matrix } A. \tag{10.24}$$

Similarly, for each $u \in U$, verify that

$$e_u^{\mathsf{T}}A \text{ is the } u\text{th row of the matrix } A. \tag{10.25}$$

Use (10.25) and (10.20) to verify that

$$\text{the inequality (10.22b) can be written as } (A^{\mathsf{T}}e_u)^{\mathsf{T}}x \leq b_u \text{ for each } u \in U. \tag{10.26}$$

10.15 Matrix-Vector Product as Linear Combination. Let $A \in \mathbb{R}^{U\times V}$ be a matrix and let $x \in \mathbb{R}^V$. Then

$$\begin{aligned}Ax &= A\Big(\sum_{v\in V} x_v e_v\Big) && \text{by (10.8)}\\ &= \sum_{v\in V} x_v(Ae_v) && \text{by (10.17c) and induction,}\end{aligned} \tag{10.27}$$

i.e., by (10.24) the vector Ax is a linear combination of the columns of A, using the entries of x as the coefficients of the linear combination.

10.16 The Rows and Columns of Incidence Matrices. Let $G \coloneqq (V, E, \varphi)$ be a loopless graph. Let $v \in V$ be a vertex and let $f \in E$ be an edge. Following (10.24) and (10.25), the fth column of the incidence matrix B_G is

$$B_G e_f = \mathbb{1}_{\varphi(f)}, \tag{10.28a}$$

and the (transpose of the) vth row of the incidence matrix B_G is:

$$B_G^{\mathsf{T}} e_v = \mathbb{1}_{\delta_G(v)}. \tag{10.28b}$$

We first prove (10.28a). One has

$$\begin{aligned}(B_G e_f)_v &= (B_G)_{vf} && \text{by (10.23)}\\ &= \big[v \in \varphi(f)\big] && \text{by (10.14)}\\ &= \mathbb{1}_{\varphi(f)}(v) && \text{by (10.10) and (10.6).}\end{aligned} \tag{10.29}$$

Since $v \in V$ is arbitrary, (10.28a) follows from (10.29).

We prove (10.28b) similarly. First note that, since G is loopless, we have

$$v \in \varphi(f) \iff f \in \delta_G(v). \tag{10.30}$$

Then

$$\begin{aligned}(B_G^{\mathsf{T}} e_v)_f &= (B_G^{\mathsf{T}})_{fv} && \text{by (10.23)}\\ &= (B_G)_{vf} && \text{by (10.18)}\\ &= [v \in \varphi(f)] && \text{by (10.14)}\\ &= [f \in \delta_G(v)] && \text{by (10.30)}\\ &= \mathbb{1}_{\delta_G(v)}(f) && \text{by (10.10) and (10.6).}\end{aligned} \tag{10.31}$$

Since $f \in E$ is arbitrary, (10.28b) follows from (10.31).

10.17 Exercise. Let $G \coloneqq (V, E, \varphi)$ be a loopless graph. Verify using (10.28a) and (10.26) that the degree constraints (9.9b) can be written compactly as $B_G x \leq \mathbb{1}$. Verify using (10.28b) and (10.26) that the linear inequalities (9.19b) can be written compactly as $B_G^\mathsf{T} y \geq \mathbb{1}$.

Lecture 11: Linear Optimization Duality

11.1 Exercise. Let $a, b, c \in \mathbb{R}^V$. Prove[i] that:

(11.1)
- (i) if $a \leq b$ and $a \geq b$, then $a = b$;
- (ii) if $a \leq b$, then $a + c \leq b + c$;
- (iii) if $a \geq 0$ and $b \geq 0$, then $a^\mathsf{T} b \geq 0$, with equality if and only if $a_i b_i = 0$ for each $i \in V$;
- (iv) if $c \geq 0$ and $a \geq b$, then[ii] $c^\mathsf{T} a \geq c^\mathsf{T} b$, with equality if and only if $a_i = b_i$ for each $i \in V$ such that $c_i > 0$.

11.2 Theorem (Linear Optimization Weak Duality). Let $A \in \mathbb{R}^{U \times V}$ be a matrix, and let $b \in \mathbb{R}^U$ and $c \in \mathbb{R}^V$ be vectors. Consider the LP (10.22) and its dual LP

$$\text{Minimize} \quad b^\mathsf{T} y \tag{11.2a}$$
$$\text{subject to} \quad y \in \mathbb{R}^U_+, \tag{11.2b}$$
$$A^\mathsf{T} y = c. \tag{11.2c}$$

If $x \in \mathbb{R}^V$ is feasible for the primal LP (10.22) and $y \in \mathbb{R}^U$ is feasible for the dual LP (11.2), then

$$c^\mathsf{T} x \leq b^\mathsf{T} y, \tag{11.3}$$

and

(11.4) equality holds in (11.3) if and only if $(b - Ax)_u = 0$ for each $u \in U$ such that $y_u > 0$.

Proof. Let $x \in \mathbb{R}^V$ be feasible for the primal LP and let $y \in \mathbb{R}^U$ be feasible for the dual LP. Then $y \geq 0$ by (11.2b) and $b \geq Ax$ by (10.22b), so (11.1)(iv) implies that

$$y^\mathsf{T} b \geq y^\mathsf{T} A x. \tag{11.5}$$

Thus,

$$\begin{aligned} c^\mathsf{T} x &= (A^\mathsf{T} y)^\mathsf{T} x && \text{by (11.2c)} \\ &= y^\mathsf{T} A x && \text{by (10.20)} \\ &\leq y^\mathsf{T} b && \text{by (11.5)} \\ &= b^\mathsf{T} y && \text{by (10.2).} \end{aligned}$$

This proves (11.3). The equality case described in (11.4) is left as an exercise[iii]. □

11.3 Linear Optimization Weak Duality. Theorem 11.2 refers to *the*[iv] dual LP of the LP (10.22). The dual LP described in §9.10 is slightly different than (11.2), since (9.18) is the dual of the slightly different LP (9.12). In this course, we will *not* go through the details about the definition of the dual of a general LP, and instead the text will display the dual LPs of many formats of (primal) LPs. Note however that the construction of the dual LP from the primal is **purely mechanical/syntactical**. Another key feature about the general construction of dual LPs[v] is that **the dual LP of the dual LP is the primal LP**.

---

[i] For (11.1)(iv), apply (11.1)(iii) using the fact that $a - b \geq 0$.
[ii] Rewrite $a \geq b$ as $a - b \geq 0$ and apply (11.1)(iii).
[iii] Use the equality case of (11.1)(iv).
[iv] If you paying a lot of attention, you will see that this definition does not make sense. Namely, (10.22) is just an optimization problem, and one cannot generally deduce the representation (in terms of a matrix and two vectors) shown in (10.22) from the feasible region and the objective function alone; there might be multiple representations. Instead, a rigorous definition with respect to representation is required to be able to refer to **the** dual LP of (10.22). We are skipping the details here just because they are beyond the scope of this course. For now let it suffice to mention that it is possible to make the definition of LP very precise in regards to representation (maybe it will be done in some appendix), and that the construction of the dual LP from the primal LP, represented in a specific form, is completely mechanical.
[v] Which we will not cover in this course.

The relation (11.3), known as **weak duality**, is a key component in understanding combinatorial optimization problems. This relation holds for all primal-dual pairs of LPs, and we will rely on this relation for other formats of primal-dual pairs of LPs, even though we only proved it for the primal-dual pair mentioned in Theorem 11.2.

Another consequence of weak duality is that (prove this!),

(11.6) if an LP is unbounded, then its dual LP is infeasible.

11.4 Linear Optimization Strong Duality. A truly remarkable and wondrous fact is that, whenever an LP and its dual LP are both feasible, they have the *same* optimal values, and both have optimal solutions. This is known[i] as **strong duality**. You should compare this to Theorem 5.5 and Corollary 8.8.

In other words, not only is weak duality an important tool in obtaining bounds[ii], but the *bounds thus obtained include the strongest bounds that can be obtained.*

Thus, the conditions from (11.4) are *necessary and sufficient* for optimality. The conditions in (11.4) are referred to as **complementary slackness**[iii].

If an LP is feasible but its dual LP is not, then it still can be proved that both have the same optimal values, though this fact should *not* be referred to as strong duality. Instead, this should be referred to as **zero duality gap**[1].

11.5 Theorem (The Fundamental Theorem of Linear Programming). Every LP is either infeasible, or unbounded, or it has an optimal solution.

11.6 Fundamental Paradigm of Combinatorial Optimization. We can now fully illustrate the fundamental paradigm applied to the Maximum Matching Problem 7.2. First, given a loopless graph $G \coloneqq (V, E, \varphi)$, the combinatorial Problem 7.2 is recast as the isomorphic/equivalent integer program

$$\begin{aligned}
&\text{(11.7a)} && \text{Maximize} && \mathbb{1}^{\mathsf{T}} x \\
&\text{(11.7b)} && \text{subject to} && B_G x \leq \mathbb{1}, \\
&\text{(11.7c)} && && x \in \mathbb{R}_+^E, \\
&\text{(11.7d)} && && x \in \mathbb{Z}^E;
\end{aligned}$$

see §9.5. Then the integrality constraint (11.7d) is dropped to obtain the LP relaxation

$$\begin{aligned}
&\text{(11.8a)} && \text{Maximize} && \mathbb{1}^{\mathsf{T}} x \\
&\text{(11.8b)} && \text{subject to} && B_G x \leq \mathbb{1}, \\
&\text{(11.8c)} && && x \in \mathbb{R}_+^E,
\end{aligned}$$

whose optimal value is $\geq \nu(G)$. Then we set up the dual LP

$$\begin{aligned}
&\text{(11.9a)} && \text{Minimize} && \mathbb{1}^{\mathsf{T}} y \\
&\text{(11.9b)} && \text{subject to} && y \in \mathbb{R}_+^V, \\
&\text{(11.9c)} && && B_G^{\mathsf{T}} y \geq \mathbb{1}.
\end{aligned}$$

And then one studies the original problem in light of this connection and all that accompanies it, e.g., the weak duality relation, the complementary slackness conditions, and so on. Insights from this understanding

[1] The term duality gap (mostly) refers to the (nonnegative) difference between the optimal values of the primal and the dual.

[i] More specifically, strong duality refers to the fact that the dual LP has an optimal solution and its objective value is the same as the optimal value of the primal LP.

[ii] Refer back to §9.9.

[iii] Again, this is beyond the scope of this course, but here is a brief hint at where the terms come from. Suppose an LP has an inequality constraint $a_i^{\mathsf{T}} x \leq b_i$. For a feasible solution x, the difference $s_i \coloneqq b_i - a_i^{\mathsf{T}} x \geq 0$ is called the *slack* of the inequality. The slack describes how loose/tight the inequality; very tight inequalities are on the verge of being violated, should the feasible solution x move a tiny little bit in a wrong direction. Note also that to a primal constraint $a_i^{\mathsf{T}} x \leq b_i$ there corresponds a dual variable y_i, which is required to be nonnegative, say. In other words, $y_i \geq 0$ is required, so one may also refer to the slack of the inequality $y_i \geq 0$. The complementary slackness conditions require that, for each pair of primal constraint and dual variable, at most one has a positive slack. That is, their slacks are complementary; if one slack is positive, the other must be zero. Note that this requirement is equivalent to $s_i y_i = 0$.

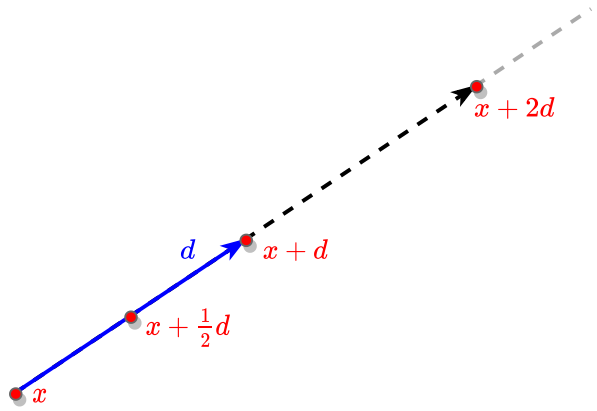


FIGURE 11.1. Line search starting from a point $x \in \mathbb{R}^V$ and moving in the direction $d \in \mathbb{R}^V$ to get to $x + \alpha d$ with stepsizes $\alpha = \frac{1}{2}, 1, 2$. The search may proceed to any stepsize $\alpha \geq 0$.

can be very helpful in designing efficient algorithms. For instance, §9.11 (see also Exercise 10.17) notes that $\mathbb{1}_K \in \mathbb{R}^V$ is feasible in (11.9) whenever $K \subseteq V$ is a vertex cover of G, so the optimal value of (11.9) is $\leq \tau(G)$.

11.7 The Line Search Principle. An important tool used in Linear Optimization[i] for designing algorithms, for proving theorems, and for developing insight, is the line search principle. Namely, start with a "good" point $x \in \mathbb{R}^V$ and a "good" search direction $d \in \mathbb{R}^V$, and consider points along the half-line $\{ x + \alpha d : \alpha \in \mathbb{R}_+ \}$ that starts at x and moves along the direction d. One may decide to choose to stop at a "step size" $\alpha \geq 0$, to end up with the point $x + \alpha d$; see, e.g., Figure 11.1. Of course, whatever "good" means depends on the context and the goal.

For example, x might be a current feasible iterate in an algorithm that solves an optimization problem $\mathcal{O}$, and the search direction d might be a direction with the property that, for small enough step sizes $\alpha \geq 0$, the point $x + \alpha d$ is feasible for $\mathcal{O}$ and with slightly better objective value than x.

Note that this principle makes sense in a continuous/geometric setting of LPs, but does not (easily) make sense in a purely combinatorial setting[ii].

11.8 Incidence Matrices of Digraphs. Let $D \coloneqq (V, A, \varphi)$ be a digraph. The $(V \times A)$ **incidence matrix** of D is the matrix $B_D \in \{0, \pm 1\}^{V \times A}$ defined by

$$B_D(va) \coloneqq \big[a \in \delta^{\text{in}}(v)\big] - \big[a \in \delta^{\text{out}}(v)\big] \qquad \text{for each } va \in V \times A. \tag{11.10}$$

See Figure 11.3 for an example.

Verify[iii] that, for every digraph $D \coloneqq (V, A, \varphi)$,

$$B_D e_a = e_v - e_u \in \mathbb{R}^V \qquad \text{for each } a \in A, \text{ where } uv \coloneqq \varphi(a). \tag{11.11}$$

11.9 Vectors of Walks. Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $W \coloneqq \langle v_0, a_1, v_1, \ldots, a_\ell, v_\ell \rangle$ be a walk in D. Denote

$$\mathbb{1}_W \coloneqq \sum_{i=1}^{\ell} e_{a_i} \in \mathbb{Z}_+^A \subseteq \mathbb{R}_+^A \tag{11.12}$$

---

[i] And even in Nonlinear Optimization!

[ii] What would be a search direction in a digraph starting from an initial walk?

[iii] We only treat the case where a is not a loop. For each $z \in V$, we have

$$\begin{aligned}
(B_D e_a)_z &= (B_D)_{za} && \text{by (10.23)} \\
&= \big[a \in \delta^{\text{in}}(z)\big] - \big[a \in \delta^{\text{out}}(z)\big] && \text{by (11.10)} \\
&= [z = v] - [z = u] && \text{since } \varphi(a) = uv \text{ and } a \text{ is not a loop} \\
&= e_z^\intercal e_v - e_z^\intercal e_u && \text{by (10.7)} \\
&= e_z^\intercal (e_v - e_u) && \text{by (10.17c)} \\
&= (e_v - e_u)_z && \text{by (10.6).}
\end{aligned}$$

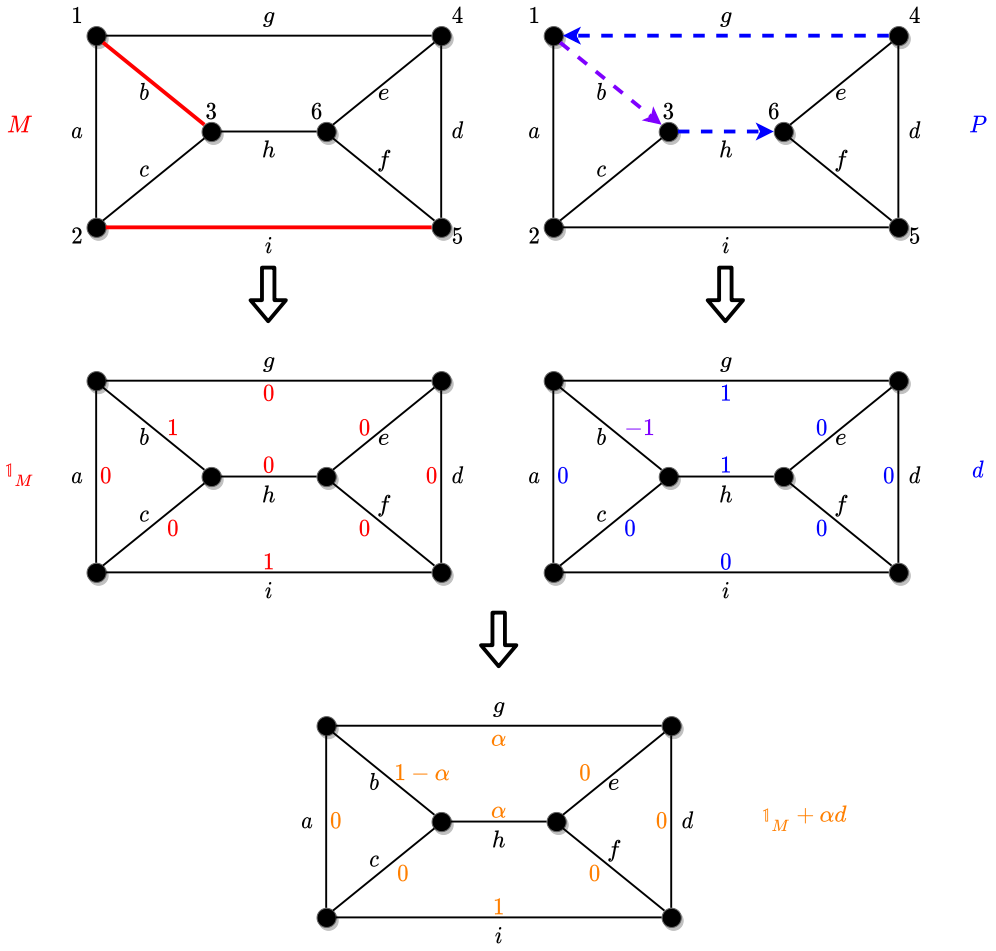
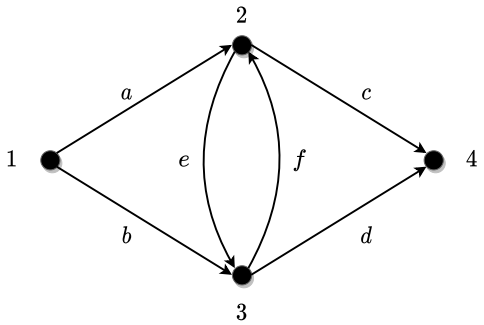


FIGURE 11.2. (More) concrete illustration of the line search principle from §11.7 on the LP relaxation (9.9) for the matching problem for the graph $G := (V, E, \varphi)$ from Figure 7.1a. The left column shows how a matching M is translated to a feasible solution $\mathbb{1}_M$ for (9.9). The right column shows how an M-augmenting path P is translated as a search direction $d \in \{0, \pm 1\}^E$. The bottom representation shows a vector $\mathbb{1}_M + \alpha d$ for stepsize $\alpha \geq 0$. What is the objective value of $\mathbb{1}_M + \alpha d$ as a function of the stepsize α, compared to the objective value of $\mathbb{1}_M$? How big can α be while keeping this vector feasible in the LP (9.9)?



$$B_D = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccccc} a & b & c & d & e & f \end{array} \\ \begin{bmatrix} -1 & -1 & 0 & 0 & 0 & 0 \\ +1 & 0 & -1 & 0 & -1 & +1 \\ 0 & +1 & 0 & -1 & +1 & -1 \\ 0 & 0 & +1 & +1 & 0 & 0 \end{bmatrix} \end{array}$$

$$\mathbb{1}_W^\intercal = \begin{array}{c} \begin{array}{cccccc} a & b & c & d & e & f \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 2 \end{bmatrix} \end{array}$$

FIGURE 11.3. A digraph $D := (V, A, \varphi)$, its incidence matrix B_D, and the incidence vector $\mathbb{1}_W$ for the walk $W := \langle 1, b, 3, f, 2, e, 3, f, 2, c, 4 \rangle$.

One may think of $\mathbb{1}_W$ as an incidence vector of the walk W, though that expression does not parse. For each $a \in A$, the ath entry of the vector $\mathbb{1}_W$ counts how many times the arc a is traversed by the walk W. See Figure 11.3 for an example.

Verify that, if $c \in \mathbb{R}^A$, then

$$c(W) = c^\mathsf{T} \mathbb{1}_W; \tag{11.13}$$

the LHS of (11.13) uses the notation set in (2.3). We claim that

$$B_D \mathbb{1}_W = e_s - e_r \qquad \text{if } W \text{ is an } rs\text{-walk.} \tag{11.14}$$

Indeed, set $W =: \langle v_0, a_1, v_1, \ldots, a_\ell, v_\ell \rangle$ and note that

$$\begin{aligned} B_D \mathbb{1}_W &= B_D \sum_{i=1}^{\ell} e_{a_i} && \text{by (11.12)} \\ &= \sum_{i=1}^{\ell} B_D e_{a_i} && \text{by (10.17c) and induction} \\ &= \sum_{i=1}^{\ell} \left(e_{v_i} - e_{v_{i-1}}\right) && \text{by (11.11)} \\ &= e_{v_\ell} - e_{v_0} && \text{by telescoping} \\ &= e_s - e_r. \end{aligned}$$

The telescoping above should remind you of (4.5).

11.10 Shortest Walks and Linear Programming. Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $r, s \in V$. The discussion from §11.9 should convince you (verify this!) that the map $W \mapsto \mathbb{1}_W$ defined on the feasible region of (SWP) is a homomorphism from (SWP) to the LP[i]

$$\begin{aligned} &\text{Minimize} && c^\mathsf{T} x && \text{(11.15a)} \\ &\text{subject to} && B_D x = e_s - e_r, && \text{(11.15b)} \\ & && x \in \mathbb{R}^A_+. && \text{(11.15c)} \end{aligned}$$

The dual LP[ii] of (11.15) is

$$\begin{aligned} &\text{Maximize} && y(s) - y(r) && \text{(11.18a)} \\ &\text{subject to} && y \in \mathbb{R}^V, && \text{(11.18b)} \\ & && B_D^\mathsf{T} y \leq c. && \text{(11.18c)} \end{aligned}$$

Note that the constraint (11.18c) involves an inequality $\leq$ in $\mathbb{R}^A$. For each arc $a \in A$, set $uv := \varphi(a)$, and the ath inequality of (11.18c) is

$$(B_D^\mathsf{T} y)_a \leq c(a). \tag{11.19}$$

The LHS of (11.19) is

$$(B_D^\mathsf{T} y)_a = e_a^\mathsf{T} B_D^\mathsf{T} y \qquad \text{by (10.6)}$$

---

[i] One might also arrive at (11.15) by first formulating an IP and then relaxing the integrality constraint. We choose not to take this path here in the interest of brevity.

[ii] If

$$\begin{aligned} &\text{Minimize} && c^\mathsf{T} x \\ &\text{subject to} && Ax = b, \\ & && x \in \mathbb{R}^A_+. \end{aligned} \tag{11.16}$$

is an LP, with $A \in \mathbb{R}^{U \times V}$ a matrix, and vectors $b \in \mathbb{R}^U$ and $c \in \mathbb{R}^V$, the dual LP is

$$\begin{aligned} &\text{Maximize} && b^\mathsf{T} y \\ &\text{subject to} && y \in \mathbb{R}^U, \\ & && A^\mathsf{T} y \leq c. \end{aligned} \tag{11.17}$$

$$\begin{aligned} &= \big(B_D e_a\big)^\mathsf{T} y && \text{by (10.20)} \\ &= \big(e_v - e_u\big)^\mathsf{T} y && \text{by (11.11)} \\ &= y(v) - y(u) && \text{by (10.17b) and (10.6).} \end{aligned}$$

Hence, the dual LP (11.18) can be written as

$$\begin{aligned} &\text{Maximize} && y(s) - y(r) && \text{(11.20a)} \\ &\text{subject to} && y \in \mathbb{R}^V, && \text{(11.20b)} \\ & && y(v) - y(u) \leq c(a) \quad \text{for each } a \in A, \text{ where } uv \coloneqq \varphi(a). && \text{(11.20c)} \end{aligned}$$

Refer back to (4.1) and compare it to (11.20c). Thus, the feasible region of the dual LP is precisely the set of feasible potentials.

11.11 Exercise. Using the context set in §11.10, verify that (SWP), (11.15), and (11.18) all have the same optimal value, by relying on Theorem 5.5.

11.12 Note that in §11.10 we did not exactly follow the Fundamental Paradigm from §11.6, in that we did *not* start with an IP formulation. In fact, it is not even clear what the integral feasible solutions to (11.15) are. It will be very beneficial to study the feasible region of (11.15) and in fact even a generalization, where we change the constraint (11.15b) to require that $B_D x$ be a *nonnegative scalar multiple*[i] of the vector $e_s - e_r$. It will take us a while to fully comprehend the leap that we are about to make.

For now we will just mention that the related optimization problems that we will see regarding this generalization are very important *models*, with a vast number of applications; see [1] or [2, Ch. 3]. They will also provide tractable *common generalizations* of the Shortest Walk Problem 2.3 and the Bipartite Matching Problem 7.11.

11.13 Flows. Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. An (r,s)**-flow** is a vector $f \in \mathbb{R}_+^A$ such that there is some scalar $\mu \in \mathbb{R}_+$ for which

$$B_D f = \mu(e_s - e_r). \tag{11.21}$$

It is easy to verify (do it) that such scalar μ is uniquely determined by f, and we call μ the **value** of the (r,s)-flow f, denoted by $\operatorname{value}(f)$.

For example, the feasible region of (11.15) is made precisely of the (r,s)-flows of value 1. Equation (11.14) shows that $\mathbb{1}_W$ is an *integral* rs-flow of value 1 whenever W is an rs-walk in D.

If $u \in \mathbb{R}_+^A$ is a vector of "capacities"[ii] on the arcs, it will be useful to consider only rs-flows f such that $f \leq u$; in this case we say that f is **feasible** (with respect to u). It will also be *extremely useful* to allow some arcs to have infinite capacity, i.e., $u_a = +\infty$.

11.14 Problem (The Maximum Flow Problem). The **maximum flow problem** is, **given**

(i) a digraph $D = (V, A, \varphi)$,
(ii) a function $u \colon A \to \mathbb{R}_+ \cup \{+\infty\}$ of "capacities" on the arcs, and
(iii) distinct vertices $r, s \in V$,

solve the LP

$$\text{(MF}(D,u,r,s)\text{)} \qquad \begin{aligned} &\text{Maximize} && \mu \\ &\text{subject to} && B_D f = \mu(e_s - e_r), \\ & && f \leq u, \\ & && (f, \mu) \in \mathbb{R}_+^A \times \mathbb{R}_+. \end{aligned}$$

Note that the inequality constraint of (MF(D,u,r,s)) is abusing the notation from §10.12, since u may take on the value $+\infty$. A more honest way to write that constraint is

$$f_a \leq u_a \qquad \text{for each } a \in A \text{ such that } u_a < +\infty. \tag{11.22}$$

The **maximum integral flow problem** is, **given**

[i] In the case of (11.15b), such scalar is the number 1.

[ii] It feel uncomfortable calling a vector of capacities by 'u' rather than 'c'. However, we reserve 'c' for vectors of costs, and 'u' may be interpreted as "upper bounds". Moreover, we will see later lower bounds for flows, and there the parallelism between 'u' for upper bounds and 'ℓ' for lower bounds will be cognitively helpful.

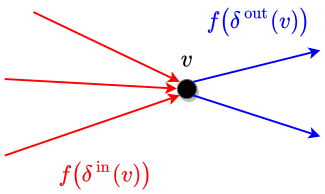


FIGURE 12.1. Illustration for the terms $f\big(\delta^{\text{in}}(v)\big)$ and $f\big(\delta^{\text{in}}(v)\big)$ for a vector $f \in \mathbb{R}^A$.

(i) a digraph $D = (V, A, \varphi)$,
(ii) a function $u\colon A \to \mathbb{Z}_+ \cup \{+\infty\}$ of *integral* "capacities" on the arcs, and
(iii) distinct vertices $r, s \in V$,

solve the optimization problem

$$\text{(MIF}(D, u, r, s)\text{)} \qquad \begin{aligned} \text{Maximize} \quad & \mu \\ \text{subject to} \quad & B_D f = \mu(e_s - e_r), \\ & f \le u, \\ & (f, \mu) \in \mathbb{Z}_+^A \times \mathbb{R}_+. \end{aligned}$$

LECTURE 12: BIPARTITE MATCHINGS AND FLOWS

12.1 Rows of Incidence Matrices. Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $v \in V$. The (transpose of the) vth row of the incidence matrix B_D is[i]

$$B_D^{\mathsf{T}} e_v = \mathbb{1}_{\delta_D^{\text{in}}(v)} - \mathbb{1}_{\delta_D^{\text{out}}(v)}. \tag{12.1}$$

Indeed, let $a \in A$. Then

$$\begin{aligned} \big(B_D^{\mathsf{T}} e_v\big)_a &= \big(B_D^{\mathsf{T}}\big)_{av} && \text{by (10.23)} \\ &= (B_D)_{va} && \text{by (10.18)} \\ &= \big[a \in \delta^{\text{in}}(v)\big] - \big[a \in \delta^{\text{out}}(v)\big] && \text{by (11.10)} \\ &= e_a^{\mathsf{T}} \mathbb{1}_{\delta_D^{\text{in}}(v)} - e_a^{\mathsf{T}} \mathbb{1}_{\delta_D^{\text{out}}(v)} && \text{by (10.10)} \\ &= e_a^{\mathsf{T}} \big(\mathbb{1}_{\delta_D^{\text{in}}(v)} - \mathbb{1}_{\delta_D^{\text{out}}(v)}\big) && \text{by (10.17c)} \\ &= \big(\mathbb{1}_{\delta_D^{\text{in}}(v)} - \mathbb{1}_{\delta_D^{\text{out}}(v)}\big)_a && \text{by (10.6)}. \end{aligned}$$

Since $a \in A$ is arbitrary, this proves (12.1).

12.2 Flows, Liquids, Excess, and Conservation. Let $D \coloneqq (V, A, \varphi)$ be a digraph. Let $f \in \mathbb{R}^A$. We want to use (12.1) to understand how to interpret the product $B_D f$, since this expression appears in the definition of flows in §11.13. For each $v \in V$, the vth entry of $B_D f \in \mathbb{R}^V$ is

$$\begin{aligned} \big(B_D f\big)_v &= e_v^{\mathsf{T}} B_D f && \text{by (10.6)} \\ &= \big(B_D^{\mathsf{T}} e_v\big)^{\mathsf{T}} f && \text{by (10.20)} \\ &= \big(\mathbb{1}_{\delta_D^{\text{in}}(v)} - \mathbb{1}_{\delta_D^{\text{out}}(v)}\big)^{\mathsf{T}} f && \text{by (12.1)} \\ &= \mathbb{1}_{\delta_D^{\text{in}}(v)}^{\mathsf{T}} f - \mathbb{1}_{\delta_D^{\text{out}}(v)}^{\mathsf{T}} f && \text{by (10.17b)} \\ &= f\big(\delta_D^{\text{in}}(v)\big) - f\big(\delta_D^{\text{out}}(v)\big) && \text{by (10.12)}, \end{aligned}$$

i.e.,

$$\big(B_D f\big)(v) = f\big(\delta_D^{\text{in}}(v)\big) - f\big(\delta_D^{\text{out}}(v)\big) \qquad \text{for each } f \in \mathbb{R}^A \text{ and each } v \in V. \tag{12.2}$$

See Figure 12.1.

[i] Verify this on the incidence matrix from Figure 11.3.

Let us interpret the digraph D as a network of pipes and junctions, e.g., vertices are junctions of pipes, and each arc is a pipe. Suppose what runs through the network is some kind of liquid, and the amount of liquid flowing through a pipe/arc a per unit of time is f_a, in whichever unit you choose. We will refer to this by saying that there are "f_a units of flow" through a. Since this is a digraph, the liquids never flow in the wrong direction, i.e., if $\varphi(a) = uv$, the flow of liquid[i] in pipe a goes from u to v.

With this interpretation in mind, the term $f\big(\delta^{\text{in}}(v)\big)$ in the RHS of (12.2) is the sum of all flows that enter the junction v, i.e.,

$$f\big(\delta^{\text{in}}(v)\big) \text{ is the flow entering } v. \tag{12.3}$$

Similarly, the term $f\big(\delta^{\text{out}}(v)\big)$ is the sum of all flows that leave the junction v, i.e.,

$$f\big(\delta^{\text{out}}(v)\big) \text{ is the flow leaving } v. \tag{12.4}$$

Hence, the difference $f(\delta^{\text{in}}(v)) - f(\delta^{\text{out}}(v))$ in the RHS of (12.2) is the net amount of flow into v, also called the *net inflow* or *excess* at v.

Given $f \in \mathbb{R}^A$, define the **excess function**

$$\text{excess}_f := B_D f \in \mathbb{R}^V. \tag{12.5}$$

By (12.2), we have

$$\text{excess}_f(v) = f\big(\delta^{\text{in}}(v)\big) - f\big(\delta^{\text{out}}(v)\big) \qquad \text{for each } f \in \mathbb{R}^A \text{ and each } v \in V. \tag{12.6}$$

In the pipe network interpretation, the excess at a vertex v accumulates at the junction v. Recall from §11.13 that, if f is an rs-flow, then

$$\text{excess}_f = B_D f = \text{value}(f) \cdot (e_s - e_r). \tag{12.7}$$

If $v \in V$ is a junction other than r and s, the vth entry of RHS of (12.7) is 0 by (10.6) and (10.7). Hence, by (12.6),

$$f\big(\delta^{\text{in}}(v)\big) = f\big(\delta^{\text{out}}(v)\big) \qquad \text{whenever } f \text{ is an } rs\text{-flow and } v \in V \setminus \{r, s\}. \tag{12.8}$$

In other words, no excess flow accumulates at junctions other than r and s in the pipe network. The condition (12.8) is known as *flow conservation*[ii].

It remains to specialize (12.7) to the rth and sth entries. By (10.7), the corresponding conditions are

$$\begin{aligned} \text{excess}_f(r) &= -\text{value}(f), \\ \text{excess}_f(s) &= \text{value}(f), \end{aligned} \qquad \text{whenever } f \text{ is an } rs\text{-flow.} \tag{12.9}$$

Recall that $\text{value}(f) \geq 0$. For the purposes of our discussion, assume that $\text{value}(f) > 0$. Then (12.9) shows that there are $\text{value}(f)$ units of excess flow at s, i.e., s is a (net) consumer of flow. Symmetrically, there are $-\text{value}(f) < 0$ units of excess flow at r, so r is a (net) producer of flow. It is usual to refer to r as the *source* of the flow and to s as the *sink*.

12.3 Bipartite Matching and Integral Flows. Let $G := (V, E, \varphi)$ be a graph, and let $U, W \subseteq V$ such that G is (U, W)-bipartite. Recall that these form an input to the Bipartite Matching Problem 7.11. We will create an instance of the Maximum Integral Flow Problem 11.14 that is equivalent to the maximum matching problem (7.1).

Let r, s be *new* vertices, i.e., elements not in V. Set $D := (V \cup \{r, s\}, A, \psi)$, where $A := U \sqcup E \sqcup W = (\{1\} \times U) \cup (\{2\} \times E) \cup (\{3\} \times W)$ and

$$\psi\big((i, e)\big) := \begin{cases} re & \text{if } i = 1 \text{ (whence } e \in U\text{)}, \\ (u, w) & \text{if } i = 2 \text{ (whence } e \in E\text{), where } \{u, w\} := \varphi(e) \text{ with } u \in U \text{ and } w \in W \\ es & \text{if } i = 3 \text{ (whence } e \in W\text{)}. \end{cases} \tag{12.10}$$

[i] You may interpret this in many ways. For instance, maybe the junction u stands higher than the junction v and the pipe is straight, so gravity ensures the direction of the flow. In other kinds of network, this interpretation may be a bit harder to fathom; perhaps liquids want to avoid getting fined with traffic violations if they run the wrong way in one-way streets.

[ii] It might remind you of Kirchhoff's current law from electricity from back in high school.

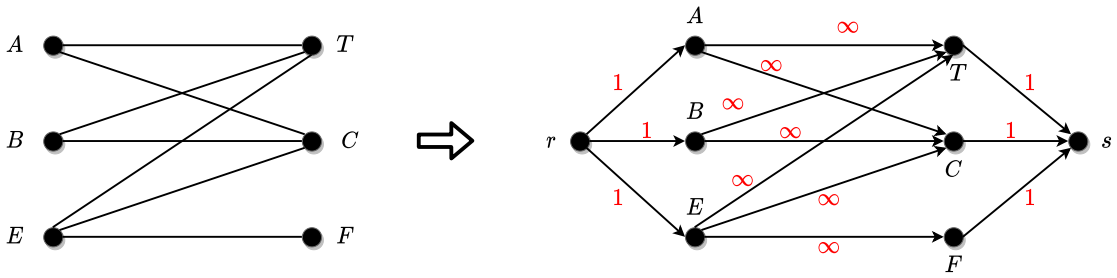


FIGURE 12.2. The (U, W)-bipartite graph G on the left, with $U \coloneqq \{A, B, E\}$ is transformed into the digraph D on the right, using the construction from §12.3. The red labels indicate the arcs capacities in $\bar{u}$. Note that the leftmost arcs with capacity 1 are $(1, A)$, $(1, B)$, and $(1, E)$.

Set[i] $\bar{u}\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ by[ii]

$$\bar{u}\big((i,e)\big) \coloneqq \begin{cases} 1 & \text{if } i = 1 \text{ or } i = 3, \\ +\infty & \text{otherwise.} \end{cases} \tag{12.11}$$

See Figure 12.2 for an example.

It is simple to verify that

$$\delta^{\text{in}}_D(r) = \varnothing \quad \text{and} \quad \delta^{\text{out}}_D(r) = \{1\} \times U, \tag{12.12a}$$

$$\delta^{\text{in}}_D(s) = \{3\} \times W \quad \text{and} \quad \delta^{\text{out}}_D(s) = \varnothing, \tag{12.12b}$$

$$\delta^{\text{in}}_D(u) = \big\{(1,u)\big\} \quad \text{and} \quad \delta^{\text{out}}_D(u) = \{2\} \times \delta_G(u) \quad \text{if } u \in U, \tag{12.12c}$$

$$\delta^{\text{in}}_D(w) = \{2\} \times \delta_G(w) \quad \text{and} \quad \delta^{\text{out}}_D(w) = \big\{(3,w)\big\} \quad \text{if } w \in W. \tag{12.12d}$$

We will (eventually[iii]) prove that

(12.13) (7.1) is equivalent to (MIF$(D, \bar{u}, r, s)$).

Let $\mathcal{M}$ be the feasible region of (7.1), i.e., $\mathcal{M}$ is the set of all matchings of G. Let $\mathcal{F}$ be the feasible region of (MIF$(D, \bar{u}, r, s)$).

12.4 In order to prove (12.13), we have to display two homomorphisms, and preferably they can be computed by efficient algorithms.

In the first homomorphism, from (7.1) to (MIF$(D, \bar{u}, r, s)$), one starts with a matching M in G and builds an integral rs-flow f_M in D. This should be intuitively simpler, since the construction of D in §12.3 is defined with this idea (of how to build the flow) in mind.

In the second homomorphism, from (MIF$(D, \bar{u}, r, s)$) to (7.1), one starts with an integral rs-flow f in D and somehow "extracts" a matching M_f from f. This seems harder, since an rs-flow is a "less constrained" object than a matching. We will have to prove properties[iv] about feasible flows to *tame* them and show that a matching can be extracted. It will be helpful to derive some more properties of flows before building this second homomorphism.

12.5 Let us discuss how the equivalence stated in (12.13) shows that the Maximum Integral Flow Problem 11.14 is at least as hard as the Bipartite Matching Problem 7.11. In other words, bipartite matching is at least as easy as maximum flows.

Assume that there is an efficient algorithm $\mathcal{A}$ for the Maximum Integral Flow Problem 11.14[v].

[i] Here we have the unfortunate conflict that we normally use u to denote both capacities of arcs and for arbitrary vertices in color class U in the bipartite graph. To avoid the conflict, we write the capacities with a bar on top.

[ii] You may wonder why we use the infinite capacity when replacing those by 1's would work. The reason is that, as a rule-of-thumb, whenever you can use an infinite capacity while preserving correctness of the formulation, you should do so. We will see some examples later.

[iii] This proof is concluded at §13.1.

[iv] After completing the first homomorphism, such properties will take us the rest of the lecture.

[v] We will build such an algorithm later in the course.

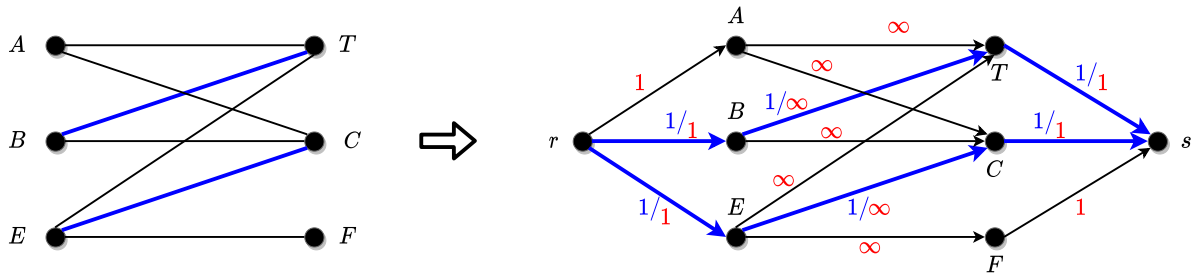


FIGURE 12.3. Expanding on Figure 12.2, this figure illustrates how the matching M on the left gets transformed to the rs-flow f_M on the right, as in §12.6. The red labels are the arc capacities from $\bar{u}$, and the blue labels indicate the amount of flow through each arc; arcs with zero flow have no blue labels to keep the picture tidy.

Let G be an input graph for the Bipartite Matching Problem 7.11. Note that there is an efficient algorithm that builds the digraph D from G as in §12.3. Now run $\mathcal{A}$ to solve (MIF$(D, \bar{u}, r, s)$) and find an optimal solution f. Finally, use an (algorithmic) homomorphism from (MIF$(D, \bar{u}, r, s)$) back to (7.1) to extract a maximum matching M_f from f.

One of the key takeaways from this course is that there are many, *many* classes of combinatorial optimization problems that are at least as easy as Maximum Integral Flow Problem 11.14.

12.6 A Homomorphism from Matchings to Flows. Restore the context set in §12.3. Let $M \in \mathcal{M}$ be a matching in G. Define $f_M \colon A \to \{0,1\}$ by

$$f_M\big((i,e)\big) := \begin{cases} [e \in V_M] & \text{if } i = 1 \text{ (whence } e \in U\text{)}, \\ [e \in M] & \text{if } i = 2 \text{ (whence } e \in E\text{)}, \\ [e \in V_M] & \text{if } i = 2 \text{ (whence } e \in W\text{)}. \end{cases} \tag{12.14}$$

We will prove that

$$f_M \text{ is an integral } rs\text{-flow of value } |M|. \tag{12.15}$$

Since $0 \le f_M \le \mathbb{1} \le \bar{u}$, this will show that $f_M \in \mathcal{F}$. See Figure 12.3 for an example. From this it will follow that

$$\text{the map } M \in \mathcal{M} \mapsto f_M \text{ is a homomorphism from (7.1) to (MIF}(D, \bar{u}, r, s)\text{).} \tag{12.16}$$

Proof. We compute excess_{f_M}. Let $u \in U$. By (12.12c), there is a single arc in D entering u, namely $(1,u)$, so by (12.14) the flow entering u is $f_M\big(\delta_D^{\text{in}}(u)\big) = [u \in V_M]$. By (12.12c), the arcs leaving u are $\{2\} \times \delta_G(u)$, and so the flow leaving u comes from the edges in $\delta_G(u)$. Hence, by (12.14), $f_M\big(\delta_D^{\text{out}}(u)\big) = \big|M \cap \delta_G(u)\big|$. Thus,

$$f_M\big(\delta_D^{\text{in}}(u)\big) = [u \in V_M] = \big|M \cap \delta_G(u)\big| = f_M\big(\delta_D^{\text{out}}(u)\big), \tag{12.17}$$

i.e., flow conservation (12.8) holds at u. A fully analogous calculation shows that flow conservation holds at every vertex $w \in W$.

By (12.14), the flow leaving r is

$$f_M\big(\delta_D^{\text{out}}(r)\big) = \sum_{u \in U} [u \in V_M] = |U \cap V_M| = |M|, \tag{12.18}$$

since G is (U,W)-bipartite. Since no arc enters r, by (12.6) the excess at r is

$$\begin{aligned} \text{excess}_{f_M}(r) &= -f_M\big(\delta_D^{\text{out}}(r)\big) && \text{by (12.6)} \\ &= -|M| && \text{by (12.18)}. \end{aligned}$$

Again by an analogous calculation, one gets $\text{excess}_{f_M}(s) = |M|$. This proves that $\text{excess}_{f_M} = |M|(e_s - e_r)$. Hence, f_M is an rs-flow of value $|M|$, and the proofs of (12.15) and (12.16) are complete. □

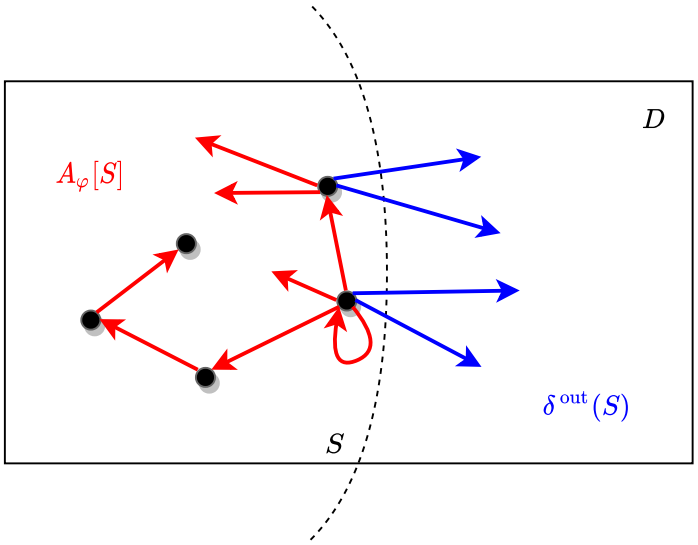


FIGURE 12.4. Illustration for the types of arcs appearing in (12.19) and (12.20).

12.7 Theorem. Let $D = (V, A, \varphi)$ be a digraph, and let $S \subseteq V$. Let $L_D(S)$ denote the set of loops at vertices of S. Then[1i]

$$\sum_{u \in S} \mathbb{1}_{\delta^{\text{out}}_D(u)} + \mathbb{1}_{L(S)} = \mathbb{1}_{A_\varphi[S]} + \mathbb{1}_{\delta^{\text{out}}_D(S)} \tag{12.19}$$

and

$$\sum_{u \in S} \mathbb{1}_{\delta^{\text{in}}_D(u)} + \mathbb{1}_{L(S)} = \mathbb{1}_{A_\varphi[S]} + \mathbb{1}_{\delta^{\text{in}}_D(S)}. \tag{12.20}$$

Proof. Let $a \in A$. Set $vw \coloneqq \varphi(a)$ and assume that $v \neq w$; the case where a is a loop is left as an exercise. Then $(\mathbb{1}_{L_D(S)})_a = e_a^\mathsf{T} \mathbb{1}_{L_D(S)} = [a \in L_D(S)] = 0$ by (10.6) and (10.10). Hence, the ath entry of the LHS of (12.19) is

$$\begin{aligned}
\Big(\sum_{u \in S} \mathbb{1}_{\delta^{\text{out}}_D(u)}\Big)_a &= e_a^\mathsf{T} \sum_{u \in S} \mathbb{1}_{\delta^{\text{out}}_D(u)} && \text{by (10.6)} \\
&= \sum_{u \in S} e_a^\mathsf{T} \mathbb{1}_{\delta^{\text{out}}_D(u)} && \text{by (10.17c) and induction} \\
&= \sum_{u \in S} \big[a \in \delta^{\text{out}}_D(u)\big] && \text{by (10.10)} \\
&= \sum_{u \in S} [v = u] && \text{since } a \text{ is a non-loop with tail } v \\
&= [v \in S] \\
&= [v \in S]\big([w \in S] + [w \notin S]\big) \\
&= [v \in S][w \in S] + [v \in S][w \notin S] \\
&= \big[a \in A_\varphi[S]\big] + \big[a \in \delta^{\text{out}}_D(S)\big] && \text{since } \varphi(a) = vw \\
&= e_a^\mathsf{T} \mathbb{1}_{A_\varphi[S]} + e_a^\mathsf{T} \mathbb{1}_{\delta^{\text{out}}_D(S)} && \text{by (10.10)} \\
&= e_a^\mathsf{T} \big(\mathbb{1}_{A_\varphi[S]} + \mathbb{1}_{\delta^{\text{out}}_D(S)}\big) && \text{by (10.17c)} \\
&= \big(\mathbb{1}_{A_\varphi[S]} + \mathbb{1}_{\delta^{\text{out}}_D(S)}\big)_a && \text{by (10.6)},
\end{aligned}$$

[1] See Figure 12.4.

[i] Recall some of the notation set in §4.7.

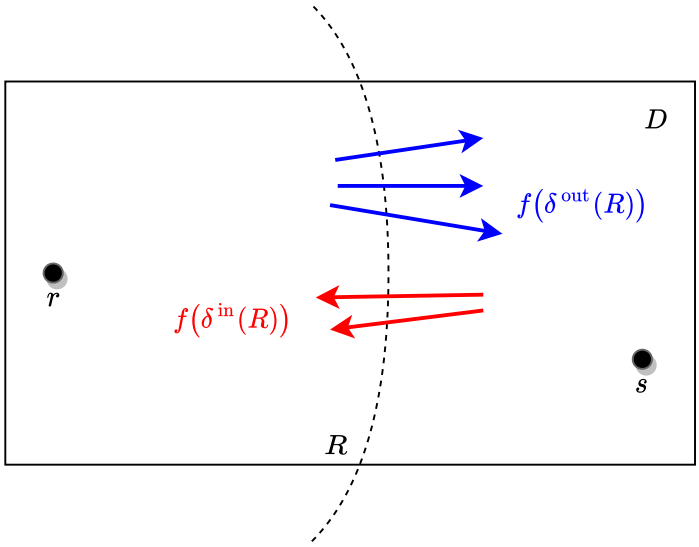


FIGURE 12.5. If f is an rs-flow and $R \subseteq V$ satisfies $r \in R$ and $s \notin R$, then (12.22) shows that the value of f is given by the total flow leaving R minus the total flow entering R.

which is the ath entry of the RHS of (12.19). This proves (12.19).

Now that (12.19) is proved, apply it to the reverse digraph D^{-1} as in (3.12) to obtain (12.20). □

12.8 Corollary (Sums of Rows of the Incidence Matrix). Let $D = (V, A, \varphi)$ be a digraph, and let $S \subseteq V$. Then

$$B_D^{\mathsf{T}} \mathbb{1}_S = \mathbb{1}_{\delta_D^{\text{in}}(S)} - \mathbb{1}_{\delta_D^{\text{out}}(S)}. \tag{12.21}$$

Proof. Let $L_D(S)$ denote the set of loops at vertices of S. We have

$$\begin{aligned}
B_D^{\mathsf{T}} \mathbb{1}_S &= B_D^{\mathsf{T}} \sum_{u \in S} e_u && \text{by (10.9)} \\
&= \sum_{u \in S} B_D^{\mathsf{T}} e_u && \text{by (10.17b) and induction} \\
&= \sum_{u \in S} \Big(\mathbb{1}_{\delta_D^{\text{in}}(u)} - \mathbb{1}_{\delta_D^{\text{out}}(u)} \Big) && \text{by (12.1)} \\
&= \sum_{u \in S} \mathbb{1}_{\delta_D^{\text{in}}(u)} - \sum_{u \in S} \mathbb{1}_{\delta_D^{\text{out}}(u)} \\
&= \Big(\mathbb{1}_{A_\varphi[S]} + \mathbb{1}_{\delta_D^{\text{in}}(S)} - \mathbb{1}_{L(S)} \Big) - \Big(\mathbb{1}_{A_\varphi[S]} + \mathbb{1}_{\delta_D^{\text{out}}(S)} - \mathbb{1}_{L(S)} \Big) && \text{by Theorem 12.7} \\
&= \mathbb{1}_{\delta_D^{\text{in}}(S)} - \mathbb{1}_{\delta_D^{\text{out}}(S)}.
\end{aligned}$$

This proves the result. □

12.9 Corollary (Flows Across Cuts). Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let f be an rs-flow, and let $R \subseteq V$ such that $r \in R$ and $s \notin R$. Then[1]

$$\text{value}(f) = -\sum_{u \in R} \text{excess}_f(u) = f\big(\delta_D^{\text{out}}(R)\big) - f\big(\delta_D^{\text{in}}(R)\big). \tag{12.22}$$

Proof. We have $\text{excess}_f(r) = -\text{value}(f)$ by (12.9). The first equation in (12.22) now follows from the fact each vertex in R other than r has excess 0. We have

$$\sum_{u \in R} \text{excess}_f(u) = \sum_{u \in R} \big(B_D f\big)(u) \qquad \text{by (12.5)}$$

[1] See Figure 12.5.

$$
\begin{aligned}
&= \sum_{u\in R} e_u^{\mathsf{T}}(B_D f) && \text{by (10.6)}\\
&= \Big(\sum_{u\in R} e_u\Big)^{\mathsf{T}} B_D f && \text{by (10.17b) and induction}\\
&= \mathbb{1}_R^{\mathsf{T}} B_D f && \text{by (10.9)}\\
&= \big(B_D^{\mathsf{T}}\mathbb{1}_R\big)^{\mathsf{T}} f && \text{by (10.20)}\\
&= \big(\mathbb{1}_{\delta_D^{\mathrm{in}}(R)} - \mathbb{1}_{\delta_D^{\mathrm{out}}(R)}\big)^{\mathsf{T}} f && \text{by (12.21)}\\
&= \mathbb{1}_{\delta_D^{\mathrm{in}}(R)}^{\mathsf{T}} f - \mathbb{1}_{\delta_D^{\mathrm{out}}(R)}^{\mathsf{T}} f && \text{by (10.17b)}\\
&= f\big(\delta_D^{\mathrm{in}}(R)\big) - f\big(\delta_D^{\mathrm{out}}(R)\big) && \text{by (10.12).}
\end{aligned}
$$

This proves the result. □

Lecture 13: Flows, Matchings, Orientations, and Directions

13.1 A Homomorphism from Flows to Matchings. We restore the context set in §12.3. Let $f \in \mathcal{F}$ be an integral rs-flow in D that is feasible with respect to $\bar{u}$. Recall that $f \in \mathbb{R}^A$ and that $A = U \sqcup E \sqcup W = (\{1\}\times U)\cup(\{2\}\times E)\cup(\{3\}\times W)$. Define

$$M_f \coloneqq \big\{\, e \in E : f_{(2,e)} > 0 \big\}. \tag{13.1}$$

We will prove that

$$M_f \text{ is a matching of size value}(f). \tag{13.2}$$

From this it will follow that

$$\text{the map } f \in \mathcal{F} \mapsto M_f \text{ is a homomorphism from (MIF}(D,\bar{u},r,s)\text{) to (7.1).} \tag{13.3}$$

Combined with (12.16), this will complete the proof of (12.13).

Proof. We first prove that

$$M_f \text{ is a matching.} \tag{13.4}$$

Let $u \in U$. The flow entering u is $f\big(\delta^{\mathrm{in}}(u)\big) = f_{(1,u)}$ by (12.12c). Since $f_{(1,u)} \le \bar{u}_{(1,u)} = 1$ and f is integral, we have $f\big(\delta^{\mathrm{in}}(u)\big) \in \{0,1\}$. By flow conservation (12.8), we must have $f\big(\delta^{\mathrm{out}}(u)\big) \in \{0,1\}$. Since $f \ge 0$ is integral, at most one arc leaving u has positive flow (equal to 1), and in fact

$$f\restriction_{\{2\}\times E} \text{ is } \{0,1\}\text{-valued.} \tag{13.5}$$

By (12.12c) and (13.4), at most one edge of $\delta_G(u)$ lies in M_f. A symmetric argument shows that, for every $w \in W$, at most one edge of $\delta_G(w)$ lies in M_f. Thus, (13.4) is proved.

It remains to show that

$$|M_f| = \text{value}(f). \tag{13.6}$$

By the definition (13.1) and by (13.5),

$$M_f = \big\{\, e \in E : f_{(2,e)} = 1 \big\} \tag{13.7}$$

Then (13.5) and (13.7) show that

$$|M_f| = \mathbb{1}_{\{2\}\times E}^{\mathsf{T}} f. \tag{13.8}$$

Set $R \coloneqq \{r\}\cup U$ and note from (12.10) that

$$\delta^{\mathrm{out}}(R) = \{2\}\times E \qquad \text{and} \qquad \delta^{\mathrm{in}}(R) = \varnothing. \tag{13.9}$$

Thus,

$$
\begin{aligned}
\text{value}(f) &= f\big(\delta^{\mathrm{out}}(R)\big) - f\big(\delta^{\mathrm{in}}(R)\big) && \text{by Corollary 12.9}\\
&= f\big(\{2\}\times E\big) && \text{by (13.9)}\\
&= \mathbb{1}_{\{2\}\times E}^{\mathsf{T}} f && \text{by (10.12)}\\
&= |M_f| && \text{by (13.8).}
\end{aligned}
$$

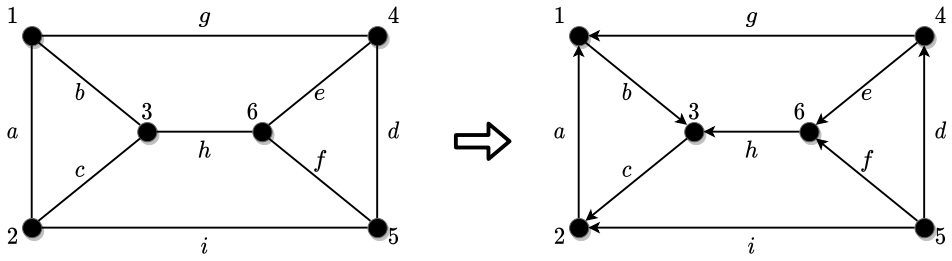


FIGURE 13.1. Example (on the right) of an orientation of the graph on the left satisfying the conditions in (13.10), with $k := 2$.

This proves (13.6). Thus, (13.2) is proved. □

13.2 Orientations with Maximum In-Degree Constraints. Consider the following problem. **Given** a graph $G := (V, E, \varphi)$ and a nonnegative integer $k \in \mathbb{N}$, **find**

(13.10) an orientation $\vec{G}$ of G such that $|\delta^{\text{in}}_{\vec{G}}(v)| \leq k$ for each $v \in V$,

if one exists. See Figure 13.1.

We will build an instance (D, u, r, s) of the Maximum Integral Flow Problem 11.14 whose optimal value is $|E|$ if and only if a desired orientation exists. Since loops play no role in this problem, we may assume that G is loopless. The idea is to "assign" each edge of G to one of its two ends, and in the orientation $\vec{G}$ this edge will enter its assigned end. Then we constrain how many edges can be assigned to the same vertex, by using the capacity function. The corresponding maximum integral flow instance will be aimed at maximizing the number of edges that get assigned to some vertex. See Figure 13.2.

Let r, s be *new* vertices, i.e., elements not in $E \sqcup V$. Define $D := \big(\{r, s\} \cup (E \sqcup V), A, \psi\big)$ and $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ where

$$E' := \{\, (e, v) : e \in E,\ v \in \varphi(e)\},$$
$$A := E \sqcup E' \sqcup V,$$
$$\psi\big((i,x)\big) := \begin{cases} (r, x) & \text{if } i = 1 \text{ (whence } x \in E), \\ x & \text{if } i = 2 \text{ (whence } x \in E'), \\ (x, s) & \text{if } i = 3 \text{ (whence } x \in V), \end{cases} \qquad \text{for every } (i,x) \in A,$$

(13.11)
$$u\big((i,x)\big) := \begin{cases} 1 & \text{if } i = 1 \text{ (whence } x \in E), \\ +\infty & \text{if } i = 2 \text{ (whence } x \in E'), \\ k & \text{if } i = 3 \text{ (whence } x \in V). \end{cases} \qquad \text{for every } (i,x) \in A.$$

We claim that[i]

(13.12) (13.10) exists if and only if the optimal value of (MIF(D, u, r, s)) is $|E|$.

Not only that, we will see that one can efficiently find a desired orientation as in (13.10) from any optimal solution for (MIF(D, u, r, s)).

13.3 From Orientations to Flows. Restore the context from §13.2. Suppose that $\vec{G}$ is an orientation of G as in (13.10). Define

(13.13)
$$f_{\vec{G}}\big((i,x)\big) := \begin{cases} 1 & \text{if } i = 1 \text{ (whence } x \in E), \\ \big[e \in \delta^{\text{in}}_{\vec{G}}(v)\big] & \text{if } i = 2 \text{ (whence } x \in E'), \text{ where } (e, v) := x \\ \big|\delta^{\text{in}}_{\vec{G}}(x)\big| & \text{if } i = 3 \text{ (whence } x \in V). \end{cases} \qquad \text{for every } (i,x) \in A.$$

(See Figure 13.3.)

Then

(13.14) $f_{\vec{G}}$ is a feasible integral rs-flow in D of value $|E|$.

[i] The proof of this is concluded at §13.7.

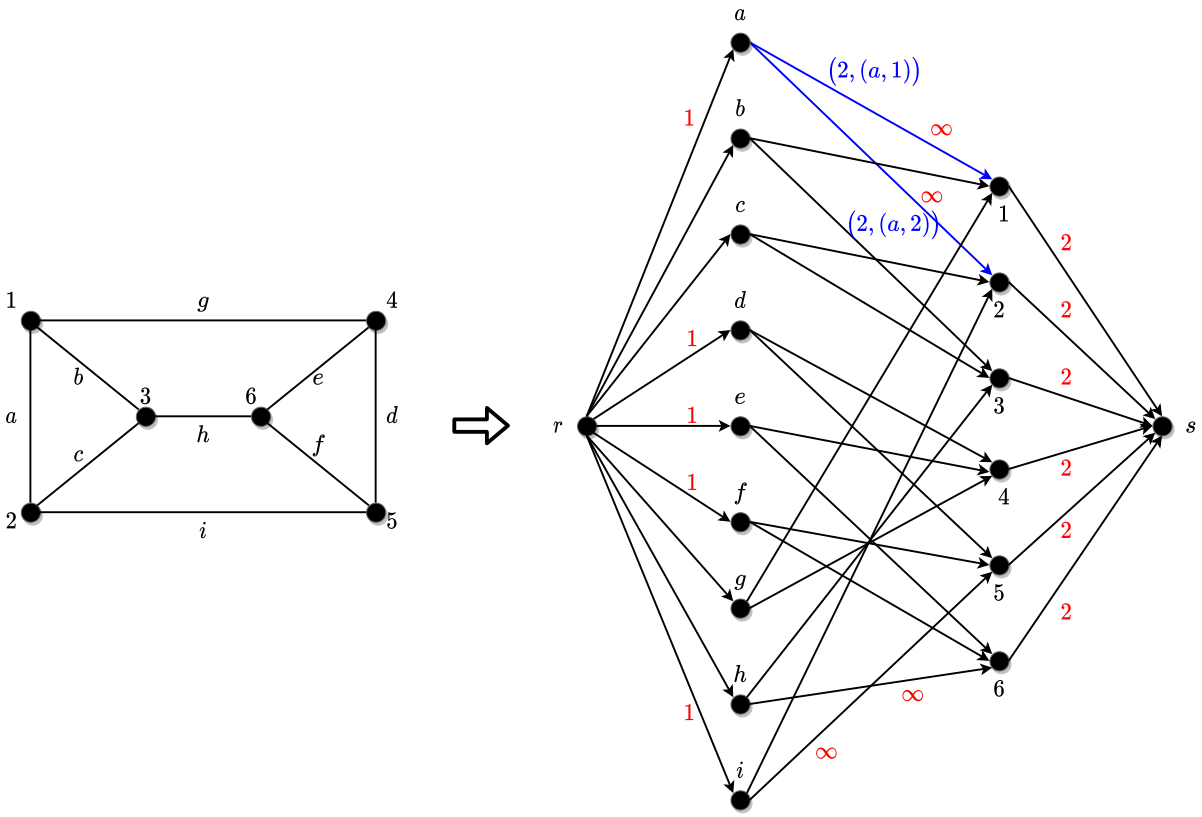


FIGURE 13.2. Instance of the Maximum Flow Problem from the orientation problem with the graph of the left as input, as specified in §13.2. The digraph on the right displays some arc capacities in red, and the labels/names of two arcs in blue.

Proof. Let us verify flow conservation (12.8). Consider a vertex of D of the form $(1, e)$, so $e \in E$ is an edge of G. Such vertex has exactly one unit of flow entering it, using the top case of (13.13). This vertex also has exactly one unit of flow leaving it, using the middle case of (13.13), since e enters exactly one vertex in the orientation $\vec{G}$. So flow conservation (12.8) holds at vertex $(1, e)$.

Consider a vertex of D of the form $(2, v)$, so $v \in V$. The set of arcs of D that enter vertex $(2, v)$ is

$$\delta_D^{\text{in}}\big((2, v)\big) = \Big\{ \big(2, (e, v)\big) : e \in \delta_G(v) \Big\}, \tag{13.15}$$

so the flow entering $(2, v)$ is

$$\begin{aligned} f_{\vec{G}}\Big(\delta_D^{\text{in}}\big((2, v)\big)\Big) &= \sum_{e \in \delta_G(v)} f_{\vec{G}}\Big(\big(2, (e, v)\big)\Big) && \text{by (10.12) and (13.15)} \\ &= \sum_{e \in \delta_G(v)} \big[e \in \delta_{\vec{G}}^{\text{in}}(v)\big] && \text{by the middle case of (13.13)} \\ &= \big|\delta_{\vec{G}}^{\text{in}}(v)\big|. \end{aligned}$$

This is exactly the flow leaving $(2, v)$, by the bottom case of (13.13). This shows that flow conservation (12.8) holds at every vertex of D other than r and s.

No arc of D enters r and the flow leaving r is clearly $|E|$ from the top case of (13.13). No arc of D leaves s and the flow entering s, from the bottom case of (13.13), is $\sum_{v \in V} \big|\delta_{\vec{G}}^{\text{in}}(v)\big| = \big|A(\vec{G})\big| = |E|$. Hence, $f_{\vec{G}}$ is an rs-flow of value $|E|$.

As for feasibility of $f_{\vec{G}}$ with respect to u, the only nontrivial verification is that $f_{\vec{G}}\big((3, v)\big) \leq u\big((3, v)\big)$ for each $v \in V$. However, this follows from the bottom case of (13.13) and (13.11), since $\vec{G}$ satisfies the conditions in (13.10). This concludes the proof of (13.14). □

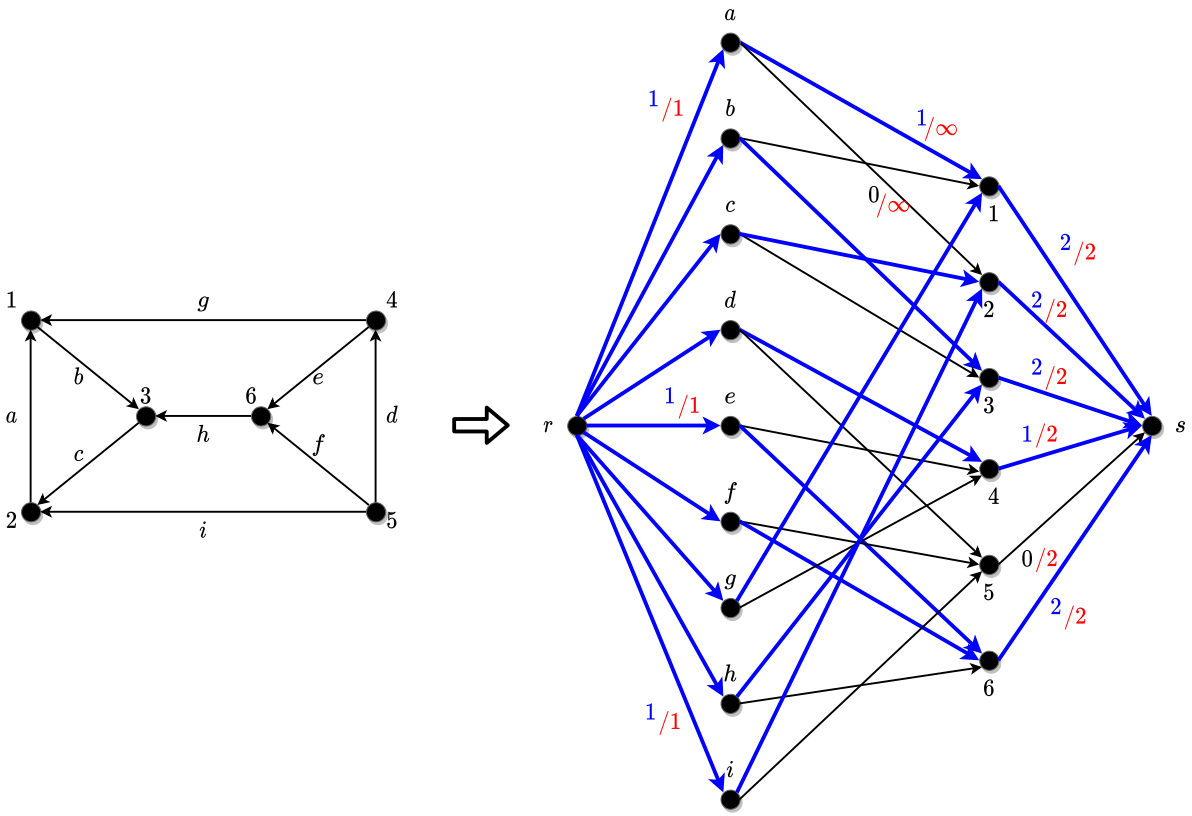


FIGURE 13.3. Example of flow $f_{\vec{G}}$ (on the right) from an orientation $\vec{G}$ (on the left) of the graph G, as defined by (13.13). As before, flows and capacities as encoded as f_a/u_a for some arcs $a \in A$, and only blue arcs carry flow. Refer back to Figure 13.2.

13.4 (r,s)**-Cuts.** Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. A set B of arcs is an (r,s)**-cut** if $B = \delta^{\text{out}}(R)$ for some $R \subseteq V$ such that $r \in R$ and $s \notin R$.

Refer back to Figure 12.5 for an example.

When there is a "capacity" function $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ defined on the arcs, it is usual to refer to $u\big(\delta_D^{\text{out}}(R)\big)$ as the *capacity* of the (r,s)-cut $\delta_D^{\text{out}}(R)$.

13.5 Theorem. Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Let $f \in \mathbb{R}^A$ be a feasible rs-flow, and let $R \subseteq V$ be such that $r \in R$ and $s \notin R$. Then

$$\text{value}(f) = f\big(\delta_D^{\text{out}}(R)\big) - f\big(\delta_D^{\text{in}}(R)\big) \leq u\big(\delta_D^{\text{out}}(R)\big), \tag{13.16}$$

and

(13.17a) equality holds in (13.16) if and only if

(13.17b) $f_a = u_a$ for each $a \in \delta_D^{\text{out}}(R)$ and

(13.17c) $f_a = 0$ for each $a \in \delta_D^{\text{in}}(R)$.

Proof. We have

$$\begin{aligned}
\text{value}(f) &= f\big(\delta^{\text{out}}(R)\big) - f\big(\delta^{\text{in}}(R)\big) && \text{by (12.22)} \\
&\leq f\big(\delta^{\text{out}}(R)\big) && \text{since } f \geq 0, \text{ with equality if and only if (13.17c) holds} \\
&\leq u\big(\delta^{\text{out}}(R)\big) && \text{since } f \leq u, \text{ with equality if and only if (13.17b) holds.}
\end{aligned}$$

This proves (13.16) and (13.17). □

13.6 Theorem 13.5 shows a weak duality relation very similar to §4.4 and §8.4 and Theorem 11.2. Namely, the value of any feasible rs-flow is upper bounded by the capacity of any rs-cut. As you might expect from this course[i], we will later prove a strong duality result involving these quantities.

Since in §13.2 the capacity of the cut $\delta_D^{\text{out}}(r)$ is $|E|$ by (13.11), it follows from Theorem 13.5 that the optimal value of (MIF(D, u, r, s)) is $\leq u\big(\delta_D^{\text{out}}(r)\big) = |E|$, so the flow $f_{\vec{G}}$ in (13.14) is optimal and thus

(13.18) the 'only if' part of (13.12) is proved.

13.7 From Flows to Orientations. Restore the context from §13.2. Let be f an integral rs-flow in D of value $|E|$, that is feasible with respect to u. We will prove below that,

$$\text{for each } e \in E, \text{ if } \{v, z\} \coloneqq \varphi(e), \text{ then } \big\{f_{(2,(e,v))}, f_{(2,(e,z))}\big\} = \{0, 1\}. \tag{13.19}$$

This implies that the incidence function ρ_f of the following orientation $\vec{G}_f$ below is properly defined: set $\vec{G}_f \coloneqq (V, E, \rho_f)$ where, for each $e \in E$ with $\{v, z\} \coloneqq \varphi(e)$, one has

$$\big[\rho_f(e) = (z, v)\big] \coloneqq f_{(2,(e,v))}. \tag{13.20}$$

Then

(13.21) the orientation $\vec{G}_f$ satisfies (13.10).

Together with (13.18), this shall conclude the proof of (13.12).

Proof Sketch. From $\text{value}(f) = |E| = u\big(\delta_D^{\text{out}}(r)\big)$ and (13.17), it follows that the flow entering any vertex of D of the form $(1, e)$ (whence $e \in E$) is 1. Hence, (13.19) follows from flow conservation (12.8) applied to $(1, e)$ and the facts that $f \geq 0$ is integral.

Abbreviate $\vec{G} \coloneqq \vec{G}_f$. Let $v \in V$. By (13.15), the flow entering $(2, v)$ is

$$\begin{aligned} f\Big(\delta_D^{\text{in}}\big((2, v)\big)\Big) &= \sum_{e \in \delta_G(v)} f_{(2,(e,v))} && \text{by (13.15)} \\ &= \sum_{e \in \delta_G(v)} \big[\rho_f(e) = (z, v)\big] && \text{by (13.20)} \\ &= \sum_{e \in \delta_G(v)} \big[e \in \delta_{\vec{G}}^{\text{in}}(v)\big] \\ &= \big|\delta_{\vec{G}}^{\text{in}}(v)\big| && \text{since } \vec{G} \text{ is an orientation of } G. \end{aligned} \tag{13.22}$$

The flow leaving $(2, v)$ is $\leq k$ by (13.11). Thus, $\big|\delta_{\vec{G}}^{\text{in}}(v)\big| \leq k$ follows from (13.22) and flow conservation (12.8) at $(2, v)$. □

13.8 Line Search and Directions of Improvement. Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Suppose $f \in \mathbb{R}^A$ is a feasible rs-flow, the current iterate in an algorithm, and that f is *not* optimal. Then there is another (feasible) rs-flow g with higher value, i.e.,

$$\text{value}(g) > \text{value}(f). \tag{13.23}$$

We want to perform line search[ii] starting from f in some direction to find another (feasible) rs-flow f' with higher value than f and start a new iteration with this next iterate, f'.

Define the direction

$$d \coloneqq g - f, \tag{13.24}$$

so that we can consider performing line search along $\{f + \alpha d : \alpha \in \mathbb{R}_+\}$ to improve the flow f. What properties are *necessary* for d to be such a direction of improvement?

Since f and g are rs-flows, we have

$$B_D f = \text{value}(f) \cdot (e_s - e_r), \tag{13.25a}$$

$$B_D g = \text{value}(g) \cdot (e_s - e_r). \tag{13.25b}$$

---

[i] Though you should *not* expect this for more general optimization problems.

[ii] Refer back to §11.7.

Hence,

$$\begin{aligned} B_D d &= B_D g - B_D f && \text{by (13.24) and (10.17c)} \\ &= \big(\text{value}(g) - \text{value}(f)\big)(e_s - e_r) && \text{by (13.25)}, \end{aligned}$$

which together with (13.23) yields the necessary condition

$$B_D d = \mu(e_s - e_r) \quad \text{for some positive } \mu \in \mathbb{R}. \tag{13.26}$$

Also, if the flow f_a through some arc $a \in A$ is at capacity, i.e.,

$$f_a = u_a, \tag{13.27}$$

then

$$\begin{aligned} d_a &= g_a - f_a && \text{by (13.24)} \\ &= g_a - u_a && \text{by (13.27)} \\ &\leq u_a - u_a = 0 && \text{since } g \leq u. \end{aligned} \tag{13.28}$$

Hence, another necessary condition is that

$$d_a \leq 0 \quad \text{for each arc } a \in A \text{ such that } f_a = u_a. \tag{13.29}$$

13.9 Vectors of Walks as Directions of Improvement. In the context of §13.8, it is not hard to see how to find a direction d of improvement using vectors of walks as in §11.9. Namely, consider the auxiliary subdigraph $D' \coloneqq (V, A', \varphi\restriction_{A'})$ where $A' \coloneqq \{\, a \in A : f_a < u_a \}$, search for a walk W from r to s in D', and take the direction $d \coloneqq \mathbb{1}_W \in \mathbb{R}^A$. Then the necessary condition (13.26) is satisfied by (11.14), and the necessary condition (13.29) holds by construction.

It is not hard to verify[i] that $f + \alpha \mathbb{1}_W$ is a feasible rs-flow of value $\text{value}(f) + \alpha$ for every sufficiently small[ii] $\alpha \geq 0$. See Figure 13.4.

Note also that the (geometric) direction d can be found "combinatorially", by searching for a walk/path in an auxiliary digraph. Thus, whenever there is an rs-walk in the auxiliary digraph D', the current rs-flow can be improved.

Lecture 14: The Max-Flow Min-Cut Theorem

14.1 (These) Vectors of Walks Are Not Enough. The idea displayed in §13.9 suggests an algorithm for continually improving feasible rs-flows. It should be asked: is that idea sufficient for developing an algorithm for solving the Maximum Flow Problem 11.14? More precisely, it is true that, whenever the current (feasible) rs-flow f is *not* optimal, then in the auxiliary digraph D' from §13.9 there is an rs-path that can be used to improve f?

Figure 14.1 shows that the answer is negative. We need to augment the search space for the search direction d in order to make this approach using directions of improvement sufficient.

14.2 More Directions of Improvement. Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let $u \colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Let f be a feasible rs-flow.

The direction of improvement d for f proposed in §13.9 satisfies $d \geq 0$, since $d = \mathbb{1}_W$ for some walk W in a digraph. However, the condition $d \geq 0$ is *not* necessary. In fact, it is easy to prove (do it!) as in (13.28) that, in the context of §13.8, another necessary condition on d is that

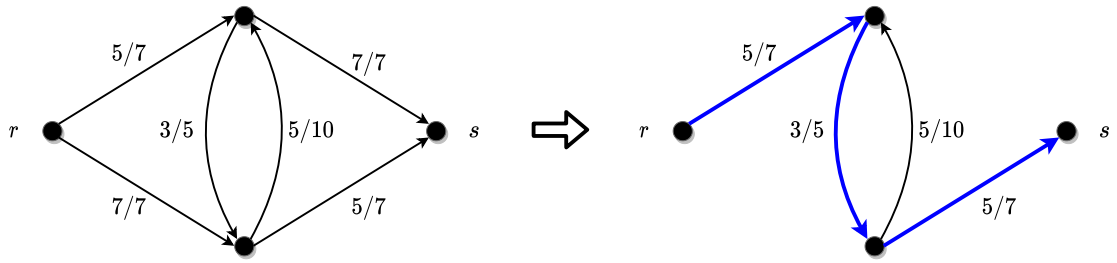
$$d_a \geq 0 \quad \text{for each arc } a \in A \text{ such that } f_a = 0. \tag{14.1}$$

In particular, note that $d_a \geq 0$ is necessary *only for arcs with zero flow*, and it is perfectly fine to have $d_a < 0$ at other arcs!

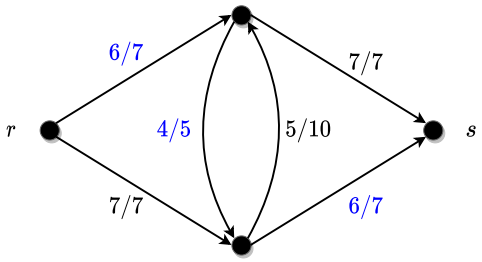
This can be used to modify/widen our search for search directions in §13.9. Namely, we form an auxiliary digraph D'' similar to D' defined there, and search for an rs-walk in D'', except that *traversing some arcs of D in the wrong direction is allowed, as long as those arcs carry a positive amount of flow.* In those arcs,

[i] Note that the conditions we verified earlier were just *necessary*, not sufficient. That is, we used the necessary conditions to "guess" a plausible direction of improvement, and later we have to verify separately that the direction really works.

[ii] Verify this claim precisely. That is, prove that there is $\varepsilon > 0$ such that, for every $0 \leq \alpha \leq \varepsilon$, the vector $f + \alpha \mathbb{1}_W$ is a feasible rs-flow of value $\text{value}(f) + \alpha$. In proving this, you should provide an *explicit* value for ε.

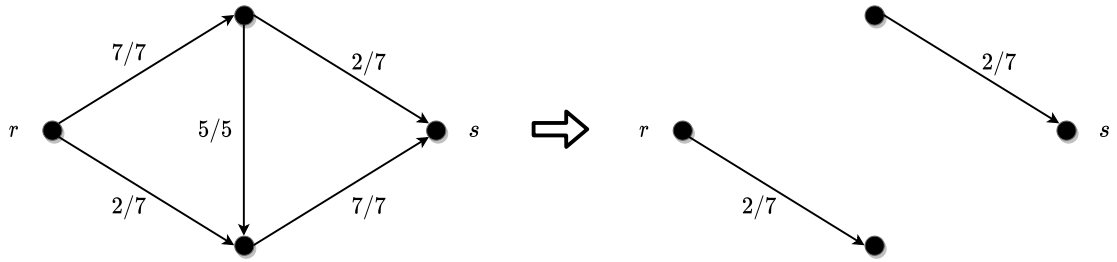


(A) An example of a feasible rs-flow f on the digraph D on the left, represented in the form f_a/u_a for each arc $a \in A$, with value$(f) = 12$. On the right, the auxiliary digraph D' as defined in §13.9.

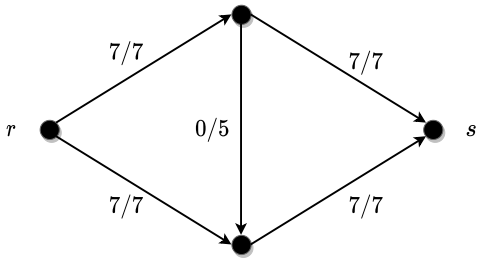


(B) A feasible rs-flow f' on the digraph D from Figure 13.4a updated from the rs-flow f there by setting $f' := f + \mathbb{1}_P$, where P is the rs-path P in the auxiliary digraph D' also from Figure 13.4a. Note that value$(f') = 13$.

FIGURE 13.4. Examples of directions of improvements for flows discussed in §13.9.

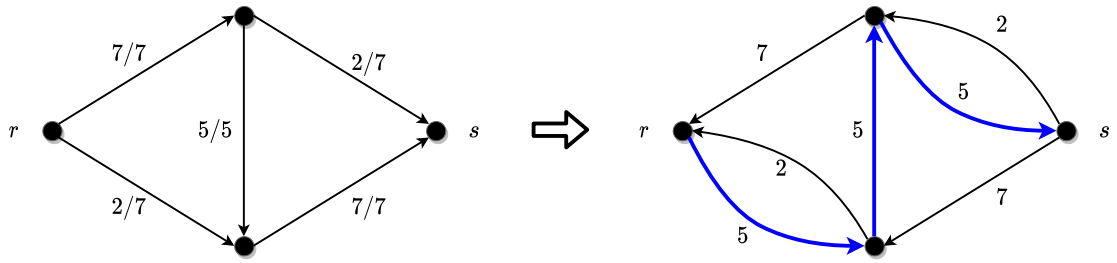


(A) An example of a feasible rs-flow f on the digraph D on the left, represented in the form f_a/u_a for each arc $a \in A$, with value$(f) = 12$. On the right, the auxiliary digraph D' as defined in §13.9.

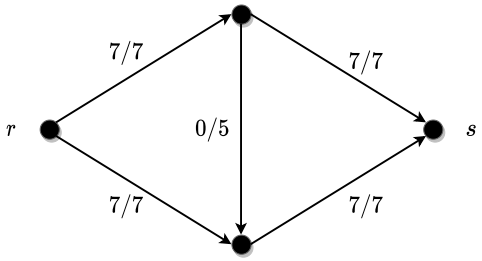


(B) A feasible rs-flow g on the digraph D from Figure 13.4a with value$(g) = 14$.

FIGURE 14.1. This example shows that there may no rs-path in the auxiliary digraph D' as defined in §13.9, even when the current (feasible) rs-flow is not optimal.



(A) An example of a feasible rs-flow f on the digraph D on the left, represented in the form f_a/u_a for each arc $a \in A$, with value$(f) = 12$. On the right, the residual digraph of f, as defined in §14.3, along with the vector u_f defined in (14.10), and an rs-path P.



(B) The feasible flow $f + 5d_P$, where f and P are shown in Figure 14.2a. Note that this is the same rs-flow displayed in Figure 14.1b.

FIGURE 14.2. Example of a flow augmentation using the residual digraph.

we will take d_a to be negative. This can be achieved by adding to D' the *reverse* of such arcs, in order to form D''.

One way to interpret these reverse/backward arcs is that one if pushing flow backward through those arcs.

14.3 Residual Digraphs. Recall (3.13) whereby we set $(v, w)^{-1} = (w, v)$ for each ordered pair (v, w). Extend that notation so that $(v, w)^{+1} := (v, w)$. Thus,

$$(v,w)^{\alpha} := \begin{cases} (v,w) & \text{if } \alpha = +1, \\ (w,v) & \text{if } \alpha = -1, \end{cases} \qquad \text{for every ordered pair } (v,w). \tag{14.2}$$

Let $D := (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Let f be a feasible rs-flow. Define the digraph

$$D(f,u) := (V, A_{f,u}, \psi), \tag{14.3}$$

where

$$A_{f,u} := \big\{ (a,+1) : a \in A,\ f_a < u_a \big\} \cup \big\{ (a,-1) : a \in A,\ f_a > 0 \big\}, \tag{14.4}$$

and

$$\psi\big((a,\alpha)\big) := \varphi(a)^{\alpha} \qquad \text{for each } (a,\alpha) \in A_{f,u}. \tag{14.5}$$

Each arc $a' \in A_{f,u}$ has the form $a' = (a, \alpha)$ where $a \in A$ and $\alpha \in \{\pm 1\}$. It is usual to refer to a' as a *forward* arc if $\alpha = +1$ and as a *backward* arc if $\alpha = -1$. The arc reversal is encoded in the sign of α, and (14.5) makes use of the notation (14.2).

The digraph $D(f, u)$ is sometimes called the **residual digraph of** f (in D, with respect to u). See Figure 14.2a.

Let $(a, \alpha) \in A_{f,u}$, so $a \in A$ and $\alpha \in \{\pm 1\}$. Set $xy := \varphi(a)$ and note that

$$B_{D(f,u)} e_{(a,\alpha)} = \begin{cases} e_y - e_x & \text{if } \alpha = +1, \\ e_x - e_y & \text{if } \alpha = -1 \end{cases}$$

by (11.11) and (14.5). This can be written more compactly[i] as

$$B_{D(f,u)}e_{(a,\alpha)} = \alpha(e_y - e_x), \tag{14.6}$$

Since $B_D e_a = e_y - e_x$ by (11.11), we can rewrite[ii] (14.6) as

$$B_{D(f,u)}e_{(a,\alpha)} = \alpha B_D e_a, \qquad \text{for each arc } (a,\alpha) \in A_{f,u}. \tag{14.7}$$

If $P \coloneqq \langle v_0, (a_1,\alpha_1), v_1, \ldots, (a_\ell,\alpha_\ell), v_\ell\rangle$ is a path in $D(f,u)$, define[iii]

$$d_P \coloneqq \sum_{i=1}^{\ell} \alpha_i e_{a_i} \in \{0,\pm 1\}^A. \tag{14.8}$$

Note that d_P can be seen as a signed version of $\mathbb{1}_P$ from (11.12).

We will soon prove that,

(14.9) if P is an rs-path in $D(f,u)$, then there is $\varepsilon > 0$ such that, for every $\lambda \in [0,\varepsilon]$, the vector $f + \lambda d_P$ is a feasible rs-flow with value $\text{value}(f) + \lambda$.

The next result, to be proved over the next paragraphs, proves a refined form[iv] of (14.9). It is illustrated by Figure 14.2.

14.4 Theorem. Let $D \coloneqq (V,A,\varphi)$ be a digraph, and let $r,s \in V$ be distinct. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Let f be a feasible rs-flow. Define $D(f,u) =: (V, A_{f,u}, \psi)$. For each arc $(a,\alpha) \in A_{f,u}$, define

$$u_f\big((a,\alpha)\big) \coloneqq \begin{cases} u_a - f_a & \text{if } \alpha = +1, \\ f_a & \text{if } \alpha = -1. \end{cases} \tag{14.10}$$

From (14.4) we get that

$$u_f(b) > 0 \quad \text{for each arc } b \in A_{f,u}. \tag{14.11}$$

Moreover,

$$u_f \text{ is integral if } f \text{ and } u \text{ are integral,} \tag{14.12}$$

where we consider a component $+\infty$ to be an integer. If P is an rs-path in the residual digraph $D(f,u)$ of f, then for each real number λ such that

$$0 \le \lambda \le \min_{b \in A(P)} u_f(b), \tag{14.13}$$

the vector $f + \lambda d_P$ is a feasible rs-flow with value $\text{value}(f) + \lambda$.

14.5 Proposition. In the context of Theorem 14.4, we have

$$B_D d_P = e_s - e_r. \tag{14.14}$$

In particular, for every $\lambda \in \mathbb{R}$, we have

$$B_D(f + \lambda d_P) = \big(\text{value}(f) + \lambda\big) \cdot (e_s - e_r). \tag{14.15}$$

Proof. Set $P =: \langle v_0, (a_1,\alpha_1), v_1, \ldots, (a_\ell,\alpha_\ell), v_\ell\rangle$. We have

$$\begin{aligned} B_D d_P &= B_D \sum_{i=1}^{\ell} \alpha_i e_{a_i} && \text{by (14.8)} \\ &= \sum_{i=1}^{\ell} \alpha_i B_D e_{a_i} && \text{by (10.17c)} \\ &= \sum_{i=1}^{\ell} B_{D(f,u)} e_{(a_i,\alpha_i)} && \text{by (14.7)} \end{aligned}$$

[i] The point here is not that we are trying to save space on the page, but rather than we are encoding case analysis as algebra.
[ii] In (14.7), there is no need to reference the ends of a, which can be an advantage over (14.6).
[iii] The fact that P is a path is crucial for the fact that all entries of d_P are either 0 or ± 1.
[iv] In the sense that it provides an explicit value for ε, which can be efficiently computed.

$$\begin{aligned} &= B_{D(f,u)} \sum_{i=1}^{\ell} e_{(a_i,\alpha_i)} && \text{by (10.17c)} \\ &= B_{D(f,u)} \mathbb{1}_P && \text{by (11.12)} \\ &= e_s - e_r && \text{by (11.14).} \end{aligned}$$

This proves (14.14). Hence, if $\lambda \in \mathbb{R}$, then

$$B_D(f + \lambda d_P) = B_D f + \lambda B_D d_P = \operatorname{value}(f) \cdot (e_s - e_r) + \lambda \cdot (e_s - e_r) = \big(\operatorname{value}(f) + \lambda\big) \cdot (e_s - e_r).$$

This proves (14.15). □

14.6 Exercise. In the context of Theorem 14.4, show that

$$d_P(a) = \big[(a,+1) \in A(P)\big] - \big[(a,-1) \in A(P)\big] \quad \text{for each } a \in A, \tag{14.16}$$

and use it to prove[i] that, for every $\lambda \in \mathbb{R}$ such that (14.13) holds, we have

$$0 \leq f + \lambda d_P \leq u. \tag{14.17}$$

14.7 We are now ready to prove Theorem 14.4. In this context, the entries of u_f are sometimes called **residual capacities**, and an rs-path in $D(f,u)$ is an **f-augmenting path**.

Proof of Theorem 14.4. Let $\lambda \in \mathbb{R}$ such that (14.13) holds. Set $g \coloneqq f + \lambda d_P$. Then $B_D g = \big(\operatorname{value}(f) + \lambda\big) \cdot (e_s - e_r)$ by Proposition 14.5, and $0 \leq g \leq u$ by Exercise 14.6. □

14.8 Problem (The Minimum (r,s)-Cut Problem). The **minimum (r,s)-cut problem** is, **given**

(i) a digraph $D = (V, A, \varphi)$,
(ii) a function $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ of "capacities" on the arcs, and
(iii) distinct vertices $r, s \in V$,

solve the optimization problem[1]

$$\begin{aligned} \text{Minimize} \quad & u\big(\delta^{\text{out}}(R)\big) \\ \text{subject to} \quad & R \subseteq V, \\ & r \in R,\ s \notin R, \\ & \{\, u_a : a \in \delta^{\text{out}}(R)\} \subseteq \mathbb{R}. \end{aligned} \tag{14.18}$$

14.9 Theorem. Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Let f be a feasible rs-flow. Set

$$R \coloneqq \{\, v \in V : r \rightsquigarrow_{D(f,u)} v\}. \tag{14.19}$$

If $s \notin R$, then

$$\operatorname{value}(f) = u\big(\delta^{\text{out}}_D(R)\big), \tag{14.20}$$

so that f is an optimal solution for (MF(D,u,r,s)) and R is an optimal solution for (14.18).

Proof. We verify that

$$\text{the equality conditions in (13.17) hold.} \tag{14.21}$$

By Exercise 3.12, we have

$$\delta^{\text{out}}_{D(f,u)}(R) = \varnothing. \tag{14.22}$$

Unpack $D(f,u) =: (V, A_{f,u}, \psi)$.

Let $a \in \delta^{\text{out}}_D(R)$. Set $xy \coloneqq \varphi(a)$. Then $x \in R$ and $y \notin R$. If $(a,+1) \in A_{f,u}$, then $\psi\big((a,+1)\big) = xy$ shows that $(a,+1) \in \delta^{\text{out}}_{D(f,u)}(R)$, which contradicts (14.22). Hence, $(a,+1) \notin A_{f,u}$. Thus, by (14.4), we have $f_a = u_a$, so (13.17b) holds.

[1] The only reason for adding the final constraint in (14.18) is that our definition of optimization problems requires the objective function to take on only real values. It is possible (and sometimes, extremely convenient) to define a more general notion of optimization problems to allow objective values to be extended real numbers. In such a context, a feasible solution for a minimization problem with objective value $+\infty$ is sometimes called "(essentially) infeasible".

[i] Split into cases according to the Iverson predicates in (14.16).

Let $a \in \delta_D^{\text{in}}(R)$. Set $yx \coloneqq \varphi(a)$. Then $x \in R$ and $y \notin R$. If $(a,-1) \in A_{f,u}$, then $\psi\big((a,-1)\big) = xy$ shows that $(a,-1) \in \delta^{\text{out}}_{D(f,u)}(R)$, which contradicts (14.22). Hence, $(a,-1) \notin A_{f,u}$. Thus, by (14.4), we have $f_a = 0$, so (13.17c) holds.

Hence, (14.21) is proved, which shows that $\text{value}(f) = u\big(\delta_D^{\text{out}}(R)\big)$. A by now familiar argument (see the proofs of Corollaries 4.3 and 8.3) shows the remainder of the result (exercise). □

14.10 Theorem (Max-Flow Min-Cut Theorem). Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Then (MF(D,u,r,s)) is either unbounded or it has an optimal solution, and the optimal values of (MF(D,u,r,s)) and (14.18) are equal, i.e., the maximum value of a feasible rs-flow is equal to the minimum capacity of an rs-cut.

Proof. Since 0 is feasible in the LP (MF(D,u,r,s)), by Theorem 11.5 that LP either is unbounded or it has an optimal solution. In the first case, (14.18) is infeasible by Theorem 13.5, so both optimal values are $+\infty$. So we may assume that

(14.23) the LP (MF(D,u,r,s)) has an optimal solution f.

Next we show that

(14.24) the residual digraph $D(f,u)$ of f has no rs-path.

Suppose for the sake of contradiction that P is an rs-path in $D(f,u)$. Set λ to be the RHS of (14.13), so that $\lambda > 0$ by (14.11). By Theorem 14.4, the vector $f + \lambda d_P$ is feasible in (MF(D,u,r,s)) with objective value greater than $\text{value}(f)$. This contradicts (14.23), and thus proves (14.24).

We now use (14.24) to verify that the hypotheses of Theorem 14.9 are met, and so (14.20) concludes the proof. □

14.11 Exercise. Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Prove that (MF(D,u,r,s)) is unbounded if and only if there is a walk W in D such that, for every $a \in A(W)$, one has $u(a) = +\infty$. Prove the same statement for (MIF(D,u,r,s)).

14.12 The Ford-Fulkerson Algorithm. By combining the ideas in Theorems 14.4 and 14.9 and Exercise 14.11, one obtains Algorithm 14.1. Verify further that the algorithm up to Line 6 can be made to run in polynomial time, and that each iteration of the **for** loop Line 7 runs in polynomial time.

14.13 Proposition. Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Suppose Algorithm 14.1 is run on input (D,u,r,s). Let $T \subseteq \mathbb{N}$ be the set of indices t for which f_t is defined. Then:

(14.25) (i) (MF(D,u,r,s)) is unbounded if and only if Line 5 returns a direction $d \in \mathbb{R}^A$ such that, for each $\lambda \in \mathbb{R}_+$, the vector λd is a feasible rs-flow of value λ;

(ii) for each $t \in T$, the vector $f_t \in \mathbb{R}^A$ is a feasible rs-flow, integral if u is integral;

(iii) the function $t \in T \mapsto \text{value}(f_t)$ is strictly increasing.

Proof Sketch. Item (14.25)(i) follows from Exercise 14.11 and (11.14).

The proof of (14.25)(ii) is a simple induction on $t \in T$. For the inductive step, note that u_{f_t} defined in Line 12 is integral by (14.12), and that λ_t defined in Line 13 is an integer. This relies on the fact that the 'min' on the RHS of Line 13 is finite. If not, by the definition (14.10) of u_{f_t}, one can extract from the path P_t in D_t an rs-path in D all of whose arcs have infinite capacity. Thus, (MF(D,u,r,s)) is unbounded by Exercise 14.11. But then Algorithm 14.1 would terminate at Line 5 by (14.25)(i).

Item (14.25)(iii) follows by the assignment in Line 14, using Theorem 14.4. □

Lecture 15: The Ford-Fulkerson Algorithm

15.1 Corollary (Termination of the Ford-Fulkerson Algorithm 14.1 For Integral Capacities). Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let $u\colon A \to \mathbb{Z}_+ \cup \{+\infty\}$ be a function. Then Algorithm 14.1 terminates, and if the optimal value μ^* of (MF(D,u,r,s)) is finite, termination occurs after at most $\mu^* + 1$ iterations of the **for** loop from Line 7.

Algorithm 14.1 Ford-Fulkerson Algorithm for the Maximum Flow Problem 11.14

Input: a tuple (D, u, r, s), formed by
(i) a digraph $D = (V, A, \varphi)$,
(ii) a function $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$,
(iii) distinct vertices r, s.

Output: if it terminates, either one of the following, each being an outcome for (MF(D, u, r, s)) and a certificate:
(i) unbounded, an rs-path P such that, for every $\lambda \in \mathbb{R}_+$, the element $(\lambda \mathbb{1}_P, \lambda)$ is feasible in (MF(D, u, r, s));
(ii) a feasible rs-flow f and a subset $R \subseteq V$ such that $r \in R \subseteq V \setminus \{s\}$, with the property that $\text{value}(f) = u\big(\delta^{\text{out}}_D(R)\big)$, as in Theorem 14.10.

```
1:  function FORD-FULKERSON(D, u, r, s)
2:      A_∞ ← { a ∈ A : u_a = +∞ }
3:      D_∞ ← (V, A_∞, φ|_{A_∞})
4:      if r ⇝_{D_∞} s then
5:          return unbounded, and 𝟙_P for any rs-path P in D_∞
6:      f_0 ← 0
7:      for t ← 0 to ∞ do
8:          D_t ← D(f_t, u), defined as in (14.3)
9:          R_t ← { v ∈ V : r ⇝_{D_t} v }
10:         if s ∈ R_t then
11:             Let P_t be an rs-path in D_t
12:             Define u_{f_t}: A(D_t) → R_+ ∪ {+∞} as in (14.10)
13:             λ_t ← min{ u_{f_t}(b) : b ∈ A(P_t) } ∈ ℝ        ▷ and λ_t ∈ ℤ if u is integral; see (14.25)(ii)
14:             f_{t+1} ← f_t + λ_t d_{P_t}
15:         else
16:             return f_t and R_t
```

Proof. If (MF(D, u, r, s)) is unbounded, (14.25)(i) shows that Algorithm 14.1 terminates at Line 5. Hence, by Theorem 11.5, we may assume that the optimal value μ^* of (MF(D, u, r, s)) is finite, and hence $\mu^* \in \mathbb{Z}$ by Theorem 14.10.

Define T as in Proposition 14.13. By (14.25)(iii), the *integer-valued* function $t \in T \mapsto \text{value}(f_t) \in \mathbb{Z}$ is strictly increasing and starts at $\text{value}(f_0) = 0$. Hence,

$$\text{value}(f_t) \geq t \qquad \text{for each } t \in T. \tag{15.1}$$

Suppose for the sake of contradiction that $\bar{t} := \mu^* + 1 \in T$. By (14.25)(ii), the vector $f_{\bar{t}}$ is a feasible solution for (MF(D, u, r, s)) with objective value $\text{value}(f_{\bar{t}}) \geq \bar{t} > \mu^*$ by (15.1), a contradiction, since μ^* is the optimal value of (MF(D, u, r, s)). Thus, for some $t \leq \mu^*$, the **if** from Line 10 tests negative, and at that point the algorithm terminates. □

15.2 Exercise. Prove Corollary 15.1 when the function u is rational-valued (except for infinite capacities), i.e., $u\colon A \to \mathbb{Q}_+ \cup \{+\infty\}$.

15.3 Theorem (Correctness of the Ford-Fulkerson Algorithm 14.1 Upon Termination). Let $D := (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Suppose Algorithm 14.1 is run on input (D, u, r, s) *and terminates* at Line 16 with $t =: T$. Then the algorithm returns an optimal solution f_T for (MF(D, u, r, s)) and an optimal solution R_T for (14.18), and f_T is integral if u is integral.

Proof. By (14.25)(ii), the vector $f_T \in \mathbb{R}^A$ is a feasible rs-flow, integral if u is integral. Since Line 16 is executed, the condition in the **if** from Line 10 is false, so the result follows from Theorem 14.9. □

15.4 Corollary (Max-Flow Integrality). Let $D := (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let $u\colon A \to \mathbb{Z}_+ \cup \{+\infty\}$ be a function. If (MF(D, u, r, s)) has an optimal solution, then it has an optimal solution that is integral, i.e., such an optimal solution is also an optimal solution for (MIF(D, u, r, s)).

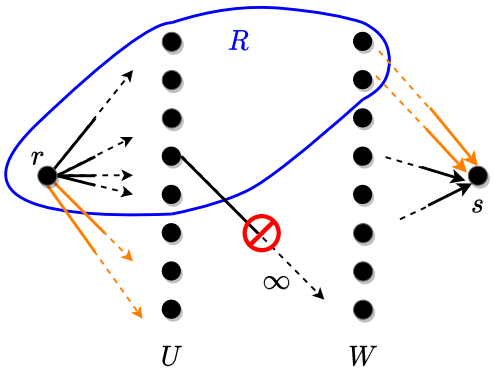


FIGURE 15.1. Schematic illustration for the proof of (15.4). The orange arcs form $\delta_D^{\text{out}}(R)$, and they go from r to $U \setminus R$, and also from $W \cap R$ to s. No arc from U to W may leave R, since those arcs have infinite capacity.

Proof. Immediate from Corollary 15.1 and Theorem 15.3. □

15.5 Theorem. Let $G \coloneqq (V, E, \varphi)$ be a (U, W)-bipartite graph. Define $(D, \bar{u}, r, s)$ as in §12.3. Recall that $A(D) = U \sqcup E \sqcup W$. Suppose that $f \in \mathbb{R}^A$ is an optimal solution for (MIF$(D, \bar{u}, r, s)$), and that $R \subseteq V(D)$ is an optimal solution for (14.18) with input $(D, \bar{u}, r, s)$. Define M_f as in (13.1) and set

$$K_R \coloneqq (U \setminus R) \cup (W \cap R). \tag{15.2}$$

Then M_f is a maximum matching of G and K_R is a minimum vertex cover of G, and

$$|M_f| = \text{value}(f) = \bar{u}\big(\delta_D^{\text{out}}(R)\big) = |K_R|. \tag{15.3}$$

Proof. The fact that M_f is a matching of G and the first equation in (15.3) follow from (13.2). The second equation in (15.3) follows from Theorem 14.10. Once we prove that

$$K_R \text{ is a vertex cover of } G \text{ of size } \bar{u}\big(\delta_D^{\text{out}}(R)\big), \tag{15.4}$$

we shall complete the proof of (15.3) and that M_f is a maximum matching and K_R a minimum vertex cover by Corollary 8.3.

Looking at Figure 15.1 may be helpful to follow the remainder of the proof. Recall that

$$A = U \sqcup E \sqcup W = (\{1\} \times U) \cup (\{2\} \times E) \cup (\{3\} \times W), \tag{15.5}$$

and note that

$$\delta^{\text{out}}(R) \cap \big(\{2\} \times E\big) = \varnothing. \tag{15.6}$$

Indeed, otherwise some arc of $\delta^{\text{out}}(R)$ has infinite capacity[i] by (12.11), whence $\bar{u}\big(\delta^{\text{out}}(R)\big) = +\infty$, which contradicts the middle equation in (15.3).

To see that

$$K_R \text{ is a vertex cover,} \tag{15.7}$$

suppose that $e \in E$ is such that $\varphi(e) = uw$, with $u \in U$ and $w \in W$, and assume that $u \notin K_R$ and $w \notin K_R$. Then $u \in U \cap R$ and $w \in W \setminus R$, so $(2, e) \in \delta^{\text{out}}(R) \cap \big(\{2\} \times E\big)$, which contradicts (15.6), and thus proves (15.7)

Next, verify from (12.10) that

$$\delta^{\text{out}}(R) \cap (\{1\} \times U) = \{1\} \times (U \setminus R), \qquad \delta^{\text{out}}(R) \cap (\{3\} \times W) = \{3\} \times (W \cap R). \tag{15.8}$$

Thus,

$$\begin{aligned}
\bar{u}\big(\delta^{\text{out}}(R)\big) &= \bar{u}\big(\delta^{\text{out}}(R) \cap (\{1\} \times U)\big) + \bar{u}\big(\delta^{\text{out}}(R) \cap (\{2\} \times E)\big) + \bar{u}\big(\delta^{\text{out}}(R) \cap (\{3\} \times W)\big) && \text{by (15.5)} \\
&= \bar{u}\big(\{1\} \times (U \setminus R)\big) + \bar{u}\big(\delta^{\text{out}}(R) \cap (\{2\} \times E)\big) + \bar{u}\big(\{3\} \times (W \cap R)\big) && \text{by (15.8)} \\
&= \bar{u}\big(\{1\} \times (U \setminus R)\big) + \bar{u}\big(\{3\} \times (W \cap R)\big) && \text{by (15.6)}
\end{aligned}$$

[i] Recall a note from §12.3 where we mentioned how helpful infinite capacities can be. This is a good example of that rule-of-thumb.

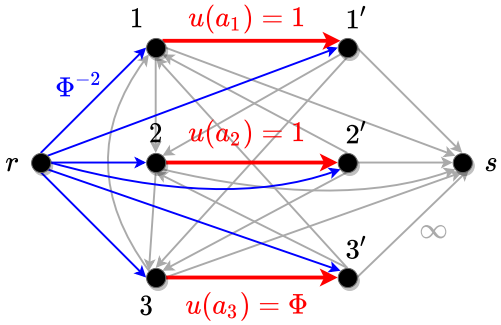


FIGURE 15.2. Input (D, u, r, s) for Exercise 15.7. The three red arcs a_1, a_2, a_3 have the capacities displayed. The blue arcs form $\delta^{\text{out}}_D(r)$, each with capacity $\Phi^{-2} = \frac{1}{1-\Phi}$, and all other gray arcs have infinite capacity.

$$
\begin{aligned}
&= \big|\{1\} \times (U \setminus R)\big| + \big|\{3\} \times (W \cap R)\big| && \text{by (12.11)} \\
&= |U \setminus R| + |W \cap R| \\
&= |K_R| && \text{by (15.2).}
\end{aligned}
$$

Together with (15.7), this concludes the proof of (15.4), and thus that of the result. □

15.6 Matchings, Flows, Cuts, Vertex Covers, Certificates, Algorithms. Theorem 15.5 and its proof contain an alternative derivation of Corollary 8.8. That any efficient algorithm for the Maximum Integral Flow Problem 11.14 could be used to solve the Bipartite Matching Problem 7.11 was already known since §12.3, using the algorithm from §13.1 and (6.6)(iv). Theorem 15.5 goes further, in that it shows that, provided that the (efficient) algorithm used for solving the Maximum Integral Flow Problem 11.14 also returns a certificate of optimality of the optimal flow found, in the form of a minimum capacity rs-cut, one can efficiently derive a minimum vertex cover from such certificate. This is transformation of certificates of optimality between different optimization problems.

The resulting algorithm appears as Algorithm 15.1. Correctness follows from Theorem 15.5.

Note the **black-box** use of the parameterized algorithm MAXINTFLOW, i.e., no detail about how the latter does its job is relevant for Algorithm 15.1. As long as MAXINTFLOW is an efficient algorithm, so is Algorithm 15.1. This illustrates the encapsulation/modularization[i] emphasized throughout this text. In particular, one could take MAXINTFLOW to be Algorithm 14.1, which satisfies the requirements by Corollary 15.1, Theorem 15.3, and Corollary 15.4.

It is amusing to verify how a call to BIPARTITE-MATCHING-VIA-FLOWS$(G, U, W,$ FORD-FULKERSON$)$ performs its job. You should notice that it executes Algorithm 8.1 precisely, via the transformations (12.14) and (13.1)!

15.7 Exercise (Failed Termination/Convergence of Algorithm 14.1 for Irrational Capacities). Consider the digraph $D := (V, A, \varphi)$, partially displayed in Figure 15.2. Let $\Phi := \frac{-1+\sqrt{5}}{2}$ be the conjugate golden ratio, i.e., $\Phi^2 + \Phi - 1 = 0$ and $0 < \Phi < 1$. The three specified arcs a_1, a_2, a_3 have capacities $u(a_1) := u(a_2) := 1$ and $u(a_3) := \Phi$. Set

$$u(a) := 2 + \Phi = \Phi^{-2} = \frac{1}{1-\Phi} = \sum_{k=0}^{\infty} \Phi^k \tag{15.9}$$

for each $a \in \delta^{\text{out}}(r)$, and let all other arcs have infinite capacity. Verify that

$$\text{there is a feasible } rs\text{-flow of value} > 2 + \Phi. \tag{15.10}$$

Now consider the following possibility for iterates f_t in Algorithm 14.1[ii]. Set $f_0 := 0$, and set $P_0 := \langle r, (r,1), 1, (1,1'), 1', (1',s), s\rangle$. Then $\lambda_0 = 1$ and the resulting flow f_1 satisfies the following properties for $t := 1$:

[i] Recall our constant nagging about proper APIs.

[ii] This description comes from [9, Sec. 10.4a]

Algorithm 15.1 Bipartite Matching via Maximum Integral Flow

Input: a tuple $(G, U, W, \textsc{MaxIntFlow})$, formed by
(i) a bipartite graph $G =: (V, E, \varphi)$,
(ii) subsets $U, W \subseteq V$ such that G is (U, W)-bipartite,
(iii) an algorithm MaxIntFlow that takes as input a tuple (D, u, r, s) for the Maximum Integral Flow Problem 11.14 and, provided (MIF(D, u, r, s)) has an optimal solution, returns an optimal solution for (MIF(D, u, r, s)) and a subset $R \subseteq V(D)$ that is an optimal solution for (14.18).
Output: a matching $M^* \subseteq E$ in and a vertex cover $K^* \subseteq V$ such that $|M^*| = |K^*|$, as in Corollary 8.3 and Theorem 15.5.

```
1:  function Bipartite-Matching-via-Flows(G, U, W, MaxIntFlow)
2:      Let r, s be new vertices, i.e., elements not in V
3:      A ← U ⊔ E ⊔ W
4:      for all u ∈ U do
5:          ψ((1, u)) ← (r, u)
6:          ū((1, u)) ← 1
7:      for all e ∈ E do
8:          Let {u, w} := φ(e), with u ∈ U and w ∈ W
9:          ψ((2, e)) ← (u, w))
10:         ū((2, e)) ← +∞
11:     for all w ∈ W do
12:         ψ((3, w)) ← (w, s)
13:         ū((3, w)) ← 1
14:     D ← (V, A, ψ)                    ▷ Lines 2 to 14 build an instance (D, ū, r, s) for MaxIntFlow
15:     (f, R) ← MaxIntFlow(D, ū, r, s)
16:     M_f ← { e ∈ E : f_(2,e) > 0 }
17:     K_R := (U \ R) ∪ (W ∩ R)
18:     return M_f and K_R
```

(15.11) (i) $\text{value}(f_t) = 1 + \Phi + \Phi^2 + \cdots + \Phi^{t-1}$,
(ii) $\{ u_{f_t}\big((a_i, +1)\big) : i \in [3] \} = \{0, \Phi^{t-1}, \Phi^t\}$,
(iii) $f_t(a) \leq 1 + \Phi + \Phi^2 + \cdots + \Phi^{t-1}$ for each $a \in A$.

The next augmenting paths P_t with $t \geq 1$ are chosen so as to preserve validity of (15.11) for each $t \in \mathbb{N}$; the exercise is to verify this by induction on t.

For $t \geq 1$, let P_t be an rs-path in D_t that traverses a_1, a_2, a_3 in the forward direction (i.e., via an arc of the form $(a_i, +1)$ in D_t), *except* for the unique[1] arc a_i with $i \in [3]$ such that $f_t(a_i) = u(a_i)$; such arc is traversed in the backwards direction. Verify that, for such rs-path P_t, one has $\lambda_t = \Phi^t$.

15.8 Golden/Freak Input Postmortem and Combinatorial Progress. Exercise 15.7 shows that, if an enemy is allowed to choose the augmenting paths P_t in Line 11, (and our enemy/user will certainly do this, when given the chance), Algorithm 14.1 *need not terminate*. Not only that, by (15.10), the feasible rs-flows maintained do not even need to converge to an optimal solution!

One may think that the source of this evil is that,

(15.12) in a strictly increasing sequence of numbers, the increase with each term may become *arbitrarily small*, i.e., there needs be no $\varepsilon > 0$ such that all terms are $\geq \varepsilon$.

This is why, in the proof of Theorem 3.1 and later, you were told to **avoid the "*and repeat*" argument for iterative procedures**[i].

You may argue that, since finite computers can only represent rational numbers exactly, Exercise 15.2 shows that the example from Exercise 15.7 is irrelevant in practice. This is a fair point. However, Exercise 15.7

[1] See (15.11)(ii).

[i] Unless you know what you are doing, of course. Naturally, you should *not* expect this to be the case.

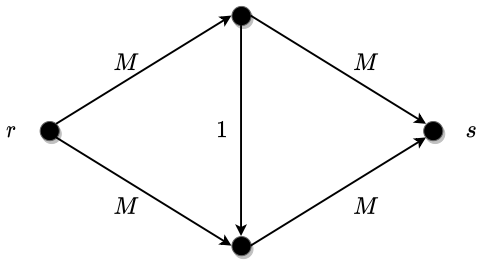


FIGURE 15.3. In this input (D, u, r, s) for Algorithm 14.1, if each augmenting path uses the vertical arc alternately in the forward and backward directions, the number of iterations is $\Theta(M)$, and yet the input has length $\Theta(\lg M)$.

shows that, e.g., in the proof of Theorem 14.10, it would *not* be sufficient to just use the "and repeat" argument. Moreover, the "root of all evil" (15.12) is not a property exclusive to real numbers; i.e., it also holds for rational numbers.

More importantly, the iteration bound from Corollary 15.1 would only ensure that Algorithm 14.1 is pseudo-polynomial; see, e.g., Figure 15.3. This is a *true* limitation of the algorithm.

We next work to transform Algorithm 14.1 into an efficient algorithm. As exposed by (15.12), even though the algorithm makes progress at each iteration, such progress can become arbitrarily small, leading to progress stall. One way around this issue is to tie in the progress to some *combinatorial quantity*, e.g., perhaps[i] for some systematic choice of augmenting paths P_t, the number of arcs in residual digraphs D_t is strictly decreasing as t increases, which would lead to an iteration bound of $2m$. We will refer to progress such as this one as *combinatorial progress*.

15.9 The Edmonds-Karp Algorithm. When Algorithm 14.1 is implemented so that the path P_t from Line 11 is always chosen to be a minimum-length rs-path in D_t, the resulting algorithm is called the Edmonds-Karp Algorithm; see Algorithm 15.2. We will show that at most nm flow augmentations are performed, and so the algorithm is guaranteed to run in polynomial time.

15.10 Let $D \coloneqq (V, A, \varphi)$ be a digraph. Define the **distance function** $\mathrm{dist}_D \colon V \times V \to \mathbb{N} \cup \{+\infty\}$ (of D) by setting $\mathrm{dist}_D(r, s)$ to be the optimal value of (SPP) with $c \coloneqq \mathbb{1}$, for every $rs \in V \times V$. Thus, $\mathrm{dist}_D(r, s)$ is the minimum length of an rs-path in D, or $+\infty$ if none exists.

15.11 Exercise. Let $D \coloneqq (V, A, \varphi)$ be a digraph. Let $r \in V$. Prove that[ii]

$$\mathrm{dist}_D(r, y) \leq \mathrm{dist}_D(r, x) + 1 \qquad \text{for each } a \in A, \text{ where } xy \coloneqq \varphi(a), \tag{15.13}$$

and that

(15.14) equality holds in (15.13) if and only if a is traversed by a minimum-length ry-path.

15.12 Exercise. Let $D = (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let $P^* \coloneqq \langle v_0, \ldots, a_\ell, v_\ell \rangle$ be a minimum-length rs-path in D. Prove that, for each $i, j \in \{0, \ldots, \ell\}$ with $i \leq j$, the path $P^*_{ij} \coloneqq \langle v_i, \ldots, a_j, v_j \rangle$ is a minimum-length (v_i, v_j)-path in D.

15.13 Exercise. Restore the context from Theorem 14.4. Let $\lambda \in \mathbb{R}$ satisfy (14.13). Set $g \coloneqq f + \lambda d_P$. Prove that,

(15.15) (i) if $(a, \alpha) \in A\big(D(g, u)\big) \setminus A\big(D(f, u)\big)$, then[iii] $(a, -\alpha) \in A(P)$;

(ii) if λ is the RHS of (14.13), there is $(a, \alpha) \in A(P)$ such that $(a, \alpha) \notin A\big(D(g, u)\big)$.

[i] This will not be the case; we are just mentioning this example here to illustrate what is meant by "combinatorial quantity".

[ii] This can be interpreted as saying that the function $\mathrm{dist}_D(r, \cdot) \colon V \to \mathbb{R}$ is a feasible potential, when restricted to the vertices where it is finite/real-valued.

[iii] Equation (14.16) might help.

Algorithm 15.2 Edmonds-Karp Algorithm for the Maximum Flow Problem 11.14

Input: a tuple (D, u, r, s), formed by
(i) a digraph $D = (V, A, \varphi)$,
(ii) a function $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$,
(iii) distinct vertices r, s.

Output: either one of the following, each being an outcome for (MF(D, u, r, s)) and a certificate:
(i) unbounded, an rs-path P such that, for every $\lambda \in \mathbb{R}_+$, the element $(\lambda \mathbb{1}_P, \lambda)$ is feasible in (MF(D, u, r, s));
(ii) a feasible rs-flow f and a subset $R \subseteq V$ such that $r \in R \subseteq V \setminus \{s\}$, with the property that value$(f) = u\big(\delta^{\text{out}}_D(R)\big)$, as in Theorem 14.10.

```
1:  function EDMONDS-KARP(D, u, r, s)
2:      A_∞ ← { a ∈ A : u_a = +∞ }
3:      D_∞ ← (V, A_∞, φ|_{A_∞})
4:      if r ⇝_{D_∞} s then
5:          return unbounded, and 𝟙_P for any rs-path P in D_∞
6:      f_0 ← 0
7:      for t ← 0 to ∞ do
8:          D_t ← D(f_t, u), defined as in (14.3)
9:          R_t ← { v ∈ V : r ⇝_{D_t} v }
10:         if s ∈ R_t then
11:             Let P_t be a minimum-length rs-path in D_t
12:             Define u_{f_t} : A(D_t) → ℝ_+ ∪ {+∞} as in (14.10)
13:             λ_t ← min{ u_{f_t}(b) : b ∈ A(P_t) } ∈ ℝ        ▷ and λ_t ∈ ℤ if u is integral; see (14.25)(ii)
14:             f_{t+1} ← f_t + λ_t d_{P_t}
15:         else
16:             return f_t and R_t
```

Lecture 16: The Edmonds-Karp Algorithm and Combinatorial Progress

16.1 Lemma. We restore the context from Theorem 14.4. Let $D := (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Let f be a feasible rs-flow. Define the function u_f of residual capacities on the set of arcs of the residual digraph $D(f, u)$ as in (14.10). Suppose that

$$P \text{ is a } \textit{minimum-length } (r, s)\text{-path in } D(f, u). \tag{16.1}$$

Set

$$\lambda := \min_{b \in A(P)} u_f(b), \tag{16.2}$$

and suppose that $\lambda \in \mathbb{R}$. Set

$$g := f + \lambda d_P. \tag{16.3}$$

Then[1]

$$\text{dist}_{D(f,u)}(r, \cdot) \leq \text{dist}_{D(g,u)}(r, \cdot), \tag{16.4a}$$

$$\text{dist}_{D(f,u)}(\cdot, s) \leq \text{dist}_{D(g,u)}(\cdot, s). \tag{16.4b}$$

Proof. We prove (16.4a). The proof of (16.4b) is analogous, and it can also be derived from (16.4a) as a black-box by considering the reverse of the digraph D, with corresponding adjustments to all the other symbols in the context.

[1] If $f\colon X \times Y \to Z$ is a function, then for each $x \in X$ the function $f(x, \cdot)\colon Y \to Z$ maps $y \in Y$ to $f(x, y)$, and similarly for the function $f(\cdot, y)\colon X \to Z$.

Suppose for the sake of contradiction that (16.4a) fails. Then the optimization problem

$$\text{Minimize} \quad \operatorname{dist}_{D(g,u)}(r,v) \tag{16.5a}$$

$$\text{subject to} \quad \operatorname{dist}_{D(g,u)}(r,v) < \operatorname{dist}_{D(f,u)}(r,v), \quad v \in V \tag{16.5b}$$

is feasible. By Theorem 1.5,

$$\text{(16.5) has an optimal solution } x. \tag{16.6}$$

From (16.6) and (16.5b), we have that $\operatorname{dist}_{D(g,u)}(r,x)$ is finite. Thus, by (2.4), there is an rx-path $Q \coloneqq \langle z_0, \ldots, (b_k,\beta_k), z_k\rangle$ in $D(g,u)$ of length $k \coloneqq \operatorname{dist}_{D(g,u)}(r,x)$. If $k=0$, then $r=x$ whence $\operatorname{dist}_{D(f,u)}(r,x) = 0 = \operatorname{dist}_{D(g,u)}(r,x)$, which contradicts feasibility of x by (16.5b). Thus, $k>0$, and we may set $y \coloneqq z_{k-1}$. Thus,

$$(b_k,\beta_k) \text{ is an arc in } D(g,u) \text{ joining } y \text{ to } x \text{ lying in a minimum-length } rx\text{-path in } D(g,u). \tag{16.7}$$

We will prove that

$$\operatorname{dist}_{D(f,u)}(r,x) > \operatorname{dist}_{D(f,u)}(r,y) + 1 \tag{16.8a}$$

and

$$(b_k,\beta_k) \notin A\big(D(f,u)\big). \tag{16.8b}$$

We shall later derive a contradiction from these facts.

By (16.7), the sufficient condition from (15.14) applied to the arc (b_k,β_k) in $D(g,u)$ holds. Thus, (15.14) shows that

$$\operatorname{dist}_{D(g,u)}(r,x) = \operatorname{dist}_{D(g,u)}(r,y) + 1. \tag{16.9}$$

Hence,

$$\operatorname{dist}_{D(g,u)}(r,y) \geq \operatorname{dist}_{D(f,u)}(r,y), \tag{16.10}$$

otherwise y would be feasible for (16.5) with (strictly) smaller objective value than the optimal solution x, by (16.9). Thus,

$$\begin{aligned} \operatorname{dist}_{D(f,u)}(r,x) &> \operatorname{dist}_{D(g,u)}(r,x) && \text{by (16.5b)} \\ &= \operatorname{dist}_{D(g,u)}(r,y) + 1 && \text{by (16.9)} \\ &\geq \operatorname{dist}_{D(f,u)}(r,y) + 1 && \text{by (16.10)}. \end{aligned}$$

This proves (16.8a).

Next, suppose that (16.8b) is false, so (b_k,β_k) is an arc of $D(f,u)$. By (14.5) and (16.7), the arc (b_k,β_k) joins y to x in $D(f,u)$. Thus, (15.13) applied to the arc (b_k,β_k) in $D(f,u)$ shows that $\operatorname{dist}_{D(f,u)}(r,x) \leq \operatorname{dist}_{D(f,u)}(r,y) + 1$, which contradicts (16.8a). This completes the proof of (16.8b).

Finally, we derive a contradiction from (16.8). Equations (16.7) and (16.8b) show that the hypotheses of (15.15)(i) for the arc (b_k,β_k) are met, whence

$$(b_k,-\beta_k) \in A(P). \tag{16.11}$$

By (16.7) and (14.5),

$$\text{the arc } (b_k,-\beta_k) \text{ joins } x \text{ to } y \text{ in } D(f,u). \tag{16.12}$$

By Exercise 15.12 and (16.1), the path in P from r to y is a minimum-length (r,y)-path in $D(f,u)$, with last arc $(b_k,-\beta_k)$ by (16.11) and (16.12). Thus, the sufficient condition from (15.14) applied to $D(f,u)$ and $(b_k,-\beta_k)$ holds, whence $\operatorname{dist}_{D(f,u)}(r,y) = \operatorname{dist}_{D(f,u)}(r,x) + 1$, which contradicts (16.8a). This concludes the proof of (16.4a). □

16.2 Exercise. Let $D \coloneqq (V,A,\varphi)$ be a digraph, let $r,s \in V$ be distinct, and let $a \in A$ lie in some minimum-length (r,s)-path in D. Let $xy \coloneqq \varphi(a)$. Apply Exercise 15.12 repeatedly to show that

$$\operatorname{dist}_D(r,y) = \operatorname{dist}_D(r,x) + 1, \tag{16.13a}$$

$$\operatorname{dist}_D(x,s) = \operatorname{dist}_D(y,s) + 1, \tag{16.13b}$$

$$\operatorname{dist}_D(r,s) = \operatorname{dist}_D(r,x) + 1 + \operatorname{dist}_D(y,s). \tag{16.13c}$$

16.3 Theorem. Restore the context from Lemma 16.1. For every feasible rs-flow h in D, define

$$\mathcal{P}_{rs}(h,u) \coloneqq \left\{ a \in A : \begin{array}{l} \text{there are } \alpha \in \{\pm 1\} \text{ and a minimum-length} \\ rs\text{-path } Q \text{ in } D(h,u) \text{ such that } (a,\alpha) \in A(Q) \end{array} \right\}. \tag{16.14}$$

If $\operatorname{dist}_{D(g,u)}(r,s) = \operatorname{dist}_{D(f,u)}(r,s)$, then[i] $\mathcal{P}_{rs}(g,u) \subsetneq \mathcal{P}_{rs}(f,u)$.

Proof. Set

$$k \coloneqq \operatorname{dist}_{D(g,u)}(r,s) = \operatorname{dist}_{D(f,u)}(r,s). \tag{16.15}$$

Let

$$a \in \mathcal{P}_{rs}(g,u). \tag{16.16}$$

Let $\alpha \in \{\pm 1\}$ so that

$$(a,\alpha) \text{ lies in a minimum-length } rs\text{-path in } D(g,u); \tag{16.17}$$

such α and path exist by (16.16). Suppose that

$$(a,\alpha) \text{ joins } x \text{ to } y \text{ in } D(g,u). \tag{16.18}$$

Then (16.17) shows that the hypotheses of Exercise 16.2 with digraph $D(g,u)$ and arc (a,α) are met. Hence,

$$\operatorname{dist}_{D(g,u)}(r,x) + \operatorname{dist}_{D(g,u)}(y,s) = k - 1, \tag{16.19}$$

where we used (16.15). From (16.19) and Lemma 16.1, we get

$$\operatorname{dist}_{D(f,u)}(r,x) + \operatorname{dist}_{D(f,u)}(y,s) \leq \operatorname{dist}_{D(g,u)}(r,x) + \operatorname{dist}_{D(g,u)}(y,s) = k - 1. \tag{16.20}$$

We will prove that

$$a \in \mathcal{P}_{rs}(f,u) \tag{16.21}$$

by splitting into the following two cases:

$$f(a) = g(a), \tag{16.22a}$$

$$f(a) \neq g(a). \tag{16.22b}$$

Let us start with (16.22a). From (16.18) we have $(a,\alpha) \in A\big(D(g,u)\big)$ and by assumption $f(a) = g(a)$, so that $(a,\alpha) \in A\big(D(f,u)\big)$ follows from (14.4). Hence, by (16.18) and (14.5),

$$(a,\alpha) \text{ joins } x \text{ to } y \text{ in } D(f,u). \tag{16.23}$$

From (16.20) we find that $\operatorname{dist}_{D(f,u)}(r,x)$ and $\operatorname{dist}_{D(f,u)}(y,s)$ are both finite. Thus, there exist a minimum-length rx-path P_{rx} in $D(f,u)$ and a minimum-length ys-path P_{ys} in $D(f,u)$. Then $P' \coloneqq P_{rx} \cdot \langle x,(a,\alpha),y\rangle \cdot P_{ys}$ is an rs-path in $D(f,u)$ by (16.23), and its length is

$$\operatorname{dist}_{D(f,u)}(r,x) + 1 + \operatorname{dist}_{D(f,u)}(y,s) \leq k = \operatorname{dist}_{D(f,u)}(r,s),$$

by (16.20), i.e., P' is a minimum-length rs-path in $D(f,u)$ that traverses (a,α), which proves (16.21).

Next we deal with case (16.22b). Since $g(a) \neq f(a)$, from (16.3) we must have $d_P(a) \neq 0$, which shows by (14.16) that $(a,+1)$ or $(a,-1)$ lies in P, and so (16.21) holds; recall (16.1).

This completes the proof of (16.21) in both cases from (16.22). Since a from (16.16) is an arbitrary arc of $\mathcal{P}_{rs}(g,u)$, this proves that

$$\mathcal{P}_{rs}(g,u) \subseteq \mathcal{P}_{rs}(f,u). \tag{16.24}$$

It remains to show that the inclusion in (16.24) is proper/strict. It suffices to display an element of the RHS of (16.24) which is not in the LHS of (16.24).

By (16.2) and (15.15)(ii) there is an arc

$$(a,-\alpha) \in A(P) \tag{16.25}$$

such that

$$(a,-\alpha) \notin A\big(D(g,u)\big). \tag{16.26}$$

[i] The notation $X \subsetneq Y$ means that $X \subseteq Y$ and $X \neq Y$; it is a more explicit notation than the loosely used '$X \subset Y$', which some authors write to denote $X \subseteq Y$. I hope those authors do not use $<$ to denote $\leq$ when comparing numbers.

From (16.25) and (16.1) we obtain that $a \in \mathcal{P}_{rs}(f,u)$. We will prove that

$$a \notin \mathcal{P}_{rs}(g,u), \tag{16.27}$$

and thus conclude that the inclusion (16.24) is strict.

Suppose for the sake of contradiction that (16.27) is false, so we may rely on the facts derived from (16.16). By (16.26), we must have $(a,\alpha) \in A\big(D(g,u)\big)$. Define $x, y \in V$ so that (16.18) holds, which implies via (14.5) that

$$(a,-\alpha) \text{ joins } y \text{ to } x \text{ in } D(f,u). \tag{16.28}$$

The hypotheses from Exercise 16.2 for digraph $D(f,u)$ and arc $(a,-\alpha)$ are met, by (16.25) and (16.1). Thus, using (16.28) and (16.15), we get

$$k = \operatorname{dist}_{D(f,u)}(r,y) + 1 + \operatorname{dist}_{D(f,u)}(x,s), \tag{16.29}$$

$$\operatorname{dist}_{D(f,u)}(r,x) = \operatorname{dist}_{D(f,u)}(r,y) + 1, \tag{16.30}$$

$$\operatorname{dist}_{D(f,u)}(y,s) = \operatorname{dist}_{D(f,u)}(x,s) + 1. \tag{16.31}$$

Thus

$$\begin{aligned}
k &= \operatorname{dist}_{D(g,u)}(r,x) + 1 + \operatorname{dist}_{D(g,u)}(y,s) && \text{by (16.19)} && (16.32)\\
&\geq \operatorname{dist}_{D(f,u)}(r,x) + 1 + \operatorname{dist}_{D(f,u)}(y,s) && \text{by Lemma 16.1} && (16.33)\\
&= \big(\operatorname{dist}_{D(f,u)}(r,y) + 1\big) + 1 + \big(\operatorname{dist}_{D(f,u)}(x,s) + 1\big) && \text{by (16.30) and (16.31)} && (16.34)\\
&= \big(\operatorname{dist}_{D(f,u)}(r,y) + 1 + \operatorname{dist}_{D(f,u)}(x,s)\big) + 2 && && (16.35)\\
&= k + 2 && \text{by (16.29)}, && (16.36)
\end{aligned}$$

a contradiction. □

16.4 Partial Orders. A **partial order** $\preccurlyeq$ on a set P is a binary relation on P if

(i) $\preccurlyeq$ is reflexive;
(ii) $\preccurlyeq$ is **antisymmetric**, i.e., $x \preccurlyeq y$ and $y \preccurlyeq x$ imply that $x = y$, for each $x, y \in P$;
(iii) $\preccurlyeq$ is transitive.

In this case, it is usual to refer to the ordered pair $(P, \preccurlyeq)$ as a **poset**, from "partially-ordered set". For elements $x, y \in P$, we say that x and y are **comparable** if $x \preccurlyeq y$ or $y \preccurlyeq x$; otherwise x and y are **incomparable**. The partial order $\preccurlyeq$ on P is **total** if any two elements of P are comparable; in this case, $\preccurlyeq$ is also called a **total order** on P. The reason partial orders are called as such is that it is generally *not* required for any two elements of P to be comparable.

16.5 Examples of Partial Orders. Examples (you should verify this!) of partial orders are

(i) the usual $\leq$ relation on the real numbers,
(ii) the inclusion relation $\subseteq$ on the power set of some set S, i.e., $(\mathcal{P}(S), \subseteq)$ is a poset, and
(iii) the subgraph relation on the collection of all subgraphs of a fixed graph G.

Note that the example in (i) is a total order, whereas the examples in items (ii) and (iii) are not, except in trivial cases (e.g., $\subseteq$ is a total order on $\mathcal{P}(\varnothing)$).

16.6 Strict Relations. If $\preccurlyeq$ is a partial order on P, the **strict relation** from $\preccurlyeq$ is the binary relation $\prec$ on P defined by

$$[x \prec y] \coloneqq [x \preccurlyeq y][x \neq y] \qquad \text{for each } x, y \in P. \tag{16.37}$$

16.7 Monotone and Strictly Monotone Functions. Let $(P, \preccurlyeq_P)$ and $(Q, \preccurlyeq_Q)$ be posets. A function $\phi\colon P \to Q$ is **monotone**[i] (with respect to $(P, \preccurlyeq_P)$ and $(Q, \preccurlyeq_Q)$) if, whenever $x, y \in P$ satisfy $x \preccurlyeq_P y$, one has $\phi(x) \preccurlyeq_Q \phi(y)$. Denote by $\prec_P$ and $\prec_Q$ the strict relations from $\preccurlyeq_P$ and $\preccurlyeq_Q$, respectively. A function $\psi\colon P \to Q$ is **strictly monotone** (with respect to $(P, \preccurlyeq_P)$ and $(Q, \preccurlyeq_Q)$) if, whenever $x, y \in P$ satisfy $x \prec_P y$, one has $\psi(x) \prec_Q \psi(y)$.

16.8 Exercise. Let $\preccurlyeq_P$ be a total order on a set P and let $(Q, \preccurlyeq_Q)$ be a poset. Prove that, if a function $\psi\colon P \to Q$ is strictly monotone, then ψ is injective.

[i] Thus, such functions are to be thought of as homomorphisms of posets.

16.9 Lexicographic Order. Let $(P, \preccurlyeq_P)$ and $(Q, \preccurlyeq_Q)$ be posets. Let $\prec_P$ denote the strict relation from $\preccurlyeq_P$. The **lexicographic order** on $P \times Q$ (with respect to $(P, \preccurlyeq_P)$ and $(Q, \preccurlyeq_Q)$) is the binary relation $\preccurlyeq_{\text{lex}}$ on $P \times Q$ defined by

$$\big[(p_1, q_1) \preccurlyeq_{\text{lex}} (p_2, q_2)\big] := [p_1 \prec_P p_2] + [p_1 = p_2][q_1 \preccurlyeq_Q q_2] \qquad \text{for each } (p_1, q_1), (p_2, q_2) \in P \times Q. \tag{16.38}$$

It is simple to verify that $\preccurlyeq_{\text{lex}}$ is a partial order on $P \times Q$, and that $\preccurlyeq_{\text{lex}}$ is total if both $\preccurlyeq_P$ and $\preccurlyeq_Q$ are total.

16.10 Corollary. Restore the context from Theorem 16.3. Set $n := |V|$ and $m := |A|$. Define $\mathcal{D} := [n-1] \cup \{\infty\}$ and let $\preccurlyeq_{\mathcal{D}}$ denote the usual order relation on $\mathcal{D}$. Define $\mathcal{A} := \{0, \ldots, m\}$, and set $[a \preccurlyeq_{\mathcal{A}} b] := [a \geq b]$ for each $a, b \in \mathcal{A}$. Let $\preccurlyeq_{\text{lex}}$ denote the **lexicographic order** on $\mathcal{D} \times \mathcal{A}$ (with respect to $(\mathcal{D}, \preccurlyeq_{\mathcal{D}})$ and $(\mathcal{A}, \preccurlyeq_{\mathcal{A}})$), as in §16.9. Let $\prec_{\text{lex}}$ denote the strict relation from $\preccurlyeq_{\text{lex}}$. For each feasible rs-flow h, define[i]

$$L(h) := \Big(\text{dist}_{D(h,u)}(r, s), \big|\mathcal{P}_{rs}(h, u)\big|\Big) \in \mathcal{D} \times \mathcal{A}. \tag{16.39}$$

Then[1]

$$L(f) \prec_{\text{lex}} L(g). \tag{16.40}$$

Proof. Immediate from Lemma 16.1 and Theorem 16.3. □

16.11 Corollary (Polynomial-Time Termination for Edmonds-Karp Algorithm). Let $D := (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let $u \colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Suppose Algorithm 15.2 is run on input (D, u, r, s), and let $T \subseteq \mathbb{N}$ be the set of indices t for which f_t is defined. Then $|T| \leq mn + 1$.

Proof. Define the symbols $\mathcal{D}$, $\mathcal{A}$, $\prec_{\text{lex}}$, L as in Corollary 16.10. We may assume that (MF(D, u, r, s)) is bounded (why?). Combine Corollary 16.10 and (14.25)(ii) to conclude that the map $\psi \colon t \in T \mapsto L(f_t)$ is strictly monotone; that (14.25)(ii) applies here follows from the fact that the iterate flows produced by Algorithm 15.2 could be produced by Algorithm 14.1. Thus, by Exercise 16.8, the function ψ is injective, whence $|T| \leq \big|\psi(T)\big|$. It is simple to verify that $\psi(T) \subseteq [n-1] \times [m] \cup \{(\infty, 0)\}$, so $|T| \leq (n-1)m + 1 \leq mn + 1$. □

16.12 Independence from the Fundamental Theorem of Linear Programming. Corollary 16.11 shows that one may now rewrite the proof of Theorem 14.10 without relying on Theorem 11.5, and replacing its use instead with Corollary 16.11.

Lecture 17: Circulations and Transshipments

17.1 Transshipments. Let $D := (V, A, \varphi)$ be a digraph. Let $b \in \mathbb{R}^V$. A vector $f \in \mathbb{R}^A_+$ is a **b-transshipment** if

$$\text{excess}_f = b. \tag{17.1}$$

Using the flow interpretation from §12.2, a vertex v is

(i) a *producer* (of flow) if $b_v < 0$, since it sends more flow outward then it receives;
(ii) a *consumer* (of flow) if $b_v > 0$.

Naturally, a vertex $v \in V$ with $b_v = 0$ must satisfy flow conservation, i.e.,

$$f\big(\delta^{\text{in}}(v)\big) = f\big(\delta^{\text{out}}(v)\big) \qquad \text{whenever } f \text{ is a } b\text{-transshipment and } v \in V \text{ is such that } b_v = 0. \tag{17.2}$$

Clearly, for distinct vertices $r, s \in V$,

$$\text{a vector } f \in \mathbb{R}^A_+ \text{ is an } rs\text{-flow of value } \mu \text{ if and only if } f \text{ is a } \mu(e_s - e_r)\text{-transshipment.} \tag{17.3}$$

Similarly as for rs-flows, if $u \colon A \to \mathbb{R}_+ \cup \{+\infty\}$ is a function of "capacities" on the arcs, we say that a b-transshipment $f \in \mathbb{R}^A_+$ is **feasible** (with respect to u) if $f \leq u$.

17.2 Proposition (Necessary Conditions for Existence of a b-Transshipment). Let $D := (V, A, \varphi)$ be a digraph, and let $b \in \mathbb{R}^V$. Let $u \colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. If there is a feasible b-transshipment, then

(i) $\mathbb{1}^{\mathsf{T}} b = 0$, and (17.4)

[1] In relation to the discussion from §15.8, the (strict) relation (16.39) is an instance of "combinatorial progress".

[i] In nonlinear and continuous optimization, some authors use the notion of a Lyapunov function (or descent function) to measure progress during execution of iterative algorithms. See, e.g., [7]. Here we are borrowing that notion informally, and the 'L' for the function that measures *combinatorial* progress is named accordingly.

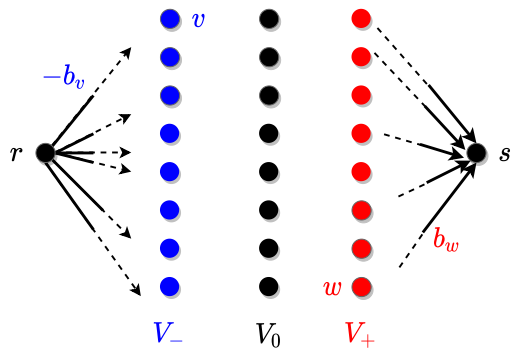


FIGURE 17.1. Illustration of the construction in Exercise 17.3.

(ii) $b(S) \leq u\big(\delta^{\text{in}}(S)\big)$ for every $S \subseteq V$.[1]

Proof. Let $f \in \mathbb{R}_+^A$ be a b-transshipment. Then

$$\begin{aligned} \mathbb{1}^\mathsf{T} b &= \mathbb{1}^\mathsf{T} \operatorname{excess}_f && \text{by (17.1)} \\ &= \mathbb{1}^\mathsf{T} B_D f && \text{by (12.5)} \\ &= (B_D^\mathsf{T} \mathbb{1})^\mathsf{T} f && \text{by (10.20)} \\ &= 0 && \text{by Corollary 12.8.} \end{aligned}$$

This proves (17.4)(i).

Suppose that $f \leq u$. Let $S \subseteq V$. Then

$$\begin{aligned} b(S) &= \mathbb{1}_S^\mathsf{T} b && \text{by (10.12)} \\ &= \mathbb{1}_S^\mathsf{T} \operatorname{excess}_f && \text{by (17.1)} \\ &= \mathbb{1}_S^\mathsf{T} B_D f && \text{by (12.5)} \\ &= (B_D^\mathsf{T} \mathbb{1}_S)^\mathsf{T} f && \text{by (10.20)} \\ &= \big(\mathbb{1}_{\delta^{\text{in}}(S)} - \mathbb{1}_{\delta^{\text{out}}(S)}\big)^\mathsf{T} f && \text{by Corollary 12.8.} \\ &= f\big(\delta^{\text{in}}(S)\big) - f\big(\delta^{\text{out}}(S)\big) && \text{by (10.17c) and (10.12)} \\ &\leq f\big(\delta^{\text{in}}(S)\big) && \text{since } f \geq 0 \\ &\leq u\big(\delta^{\text{in}}(S)\big) && \text{since } f \leq u. \end{aligned}$$

This proves (17.4)(ii). □

17.3 Exercise. Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $b \in \mathbb{R}^V$. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. (This reduction can be followed with Figure 17.1.) Let r, s be *new* vertices, i.e., elements not in V. Let

$$V_+ \coloneqq \{\, v \in V : b_v > 0\}, \tag{17.5a}$$

$$V_- \coloneqq \{\, v \in V : b_v < 0\}. \tag{17.5b}$$

Define the digraph $D' \coloneqq (V', A', \varphi')$, where $V' \coloneqq \{r, s\} \cup V$, $A' \coloneqq V_- \sqcup A \sqcup V_+$ and

$$\varphi'\big((i,x)\big) \coloneqq \begin{cases} (r, x) & \text{if } i = 1 \text{ (whence } x \in V_-), \\ \varphi(x) & \text{if } i = 2 \text{ (whence } x \in A), \\ (x, s) & \text{if } i = 3 \text{ (whence } x \in V_+), \end{cases} \qquad \text{for each } (i,x) \in A'.$$

[1] One way to interpret this is that $b(S)$ is the "demand" that the set S wants to "consume", and if the condition is violated by some set S, i.e., $b(S) > u(\delta^{\text{in}}(S))$, then this hungry set S requires more consumption than the maximum capacity $u(\delta^{\text{in}}(S))$ that can enter its mouth.

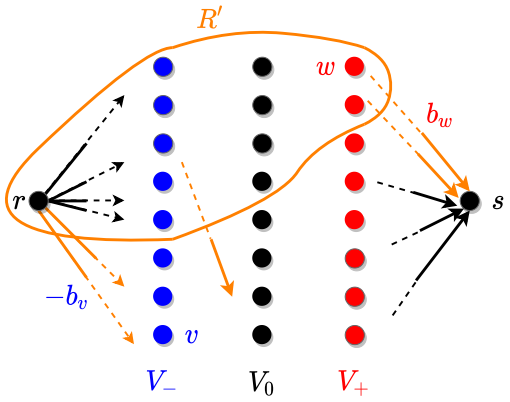


FIGURE 17.2. Illustration for the proof of Proposition 17.4. The orange arcs form $\delta^{\text{out}}_{D'}(R')$.

Define $u' \colon A' \to \mathbb{R}_+ \cup \{+\infty\}$ by

$$u'\big((i,x)\big) := \begin{cases} -b_x & \text{if } i = 1 \text{ (whence } x \in V_-), \\ u(x) & \text{if } i = 2 \text{ (whence } x \in A), \\ +b_x & \text{if } i = 3 \text{ (whence } x \in V_+), \end{cases} \qquad \text{for each } (i,x) \in A'. \tag{17.6}$$

Define $\psi' \colon \mathbb{R}^A \to \mathbb{R}^{A'}$ by

$$\big(\psi'(f)\big)_{(i,x)} := \begin{cases} -b_x & \text{if } i = 1 \text{ (whence } x \in V_-), \\ f(x) & \text{if } i = 2 \text{ (whence } x \in A), \\ +b_x & \text{if } i = 3 \text{ (whence } x \in V_+), \end{cases} \qquad \text{for each } f \in \mathbb{R}^A \text{ and } (i,x) \in A'. \tag{17.7}$$

Define $\psi \colon \mathbb{R}^{A'} \to \mathbb{R}^A$ by

$$\big(\psi(f')\big)(a) := f'\big((2,a)\big), \qquad \text{for each } f' \in \mathbb{R}^{A'} \text{ and } a \in A. \tag{17.8}$$

Prove that:

(17.9) (i) if $f \in \mathbb{R}^A_+$ is a b-transshipment, feasible with respect to u, then $f' := \psi'(f)$ is an rs-flow in D', feasible with respect to u', of value $b(V_+)$; moreover, f' is integral if b and f are integral;

(ii) if $f' \in \mathbb{R}^{A'}$ is an rs-flow in D', feasible with respect to u', of value[i] $b(V_+)$, then $f := \psi(f')$ is a b-transshipment, feasible with respect to u; moreover, f is integral if f' is integral.

17.4 Proposition. Restore the context from Exercise 17.3. If $R' \subseteq V'$ is such that $r \in R'$, $s \notin R'$, and

$$u'\big(\delta^{\text{out}}_{D'}(R')\big) < b(V_+), \tag{17.10}$$

then $S := V \setminus R'$ satisfies $b(S) > u\big(\delta^{\text{in}}_D(S)\big)$.

Proof. Set $R := R' \setminus \{r\} \subseteq V$. (The proof can be followed through Figure 17.2.) It is simple to verify that

$$\delta^{\text{out}}_{D'}(R') = (V_- \setminus R) \sqcup \delta^{\text{out}}_D(R) \sqcup (V_+ \cap R). \tag{17.11}$$

Hence, by (17.6),

$$u'\big(\delta^{\text{out}}_{D'}(R')\big) = -b(V_- \setminus R) + u\big(\delta^{\text{out}}_D(R)\big) + b(V_+ \cap R). \tag{17.12}$$

By (17.10) and using that $S = V \setminus R$, we find

$$\begin{aligned} b(V_+) &> -b(V_- \setminus R) + u\big(\delta^{\text{out}}_D(R)\big) + b(V_+ \cap R) \\ &\Longrightarrow\ b(V_+) - b(V_+ \cap R) + b(V_- \setminus R) > u\big(\delta^{\text{out}}_D(R)\big) \\ &\Longrightarrow\ b(V_+ \setminus R) + b(V_- \setminus R) > u\big(\delta^{\text{out}}_D(R)\big) \\ &\Longrightarrow\ b(S) > u\big(\delta^{\text{in}}_D(S)\big). \end{aligned}$$

The final implication uses the fact that vertices of V outside $V_+ \cup V_-$ satisfy $b_v = 0$. □

[i] To use this fact in your proof, mimic the argument from the beginning of the proof in §13.7 that relies on (13.17).

Algorithm 17.1 Transshipment via Maximum Flow

Input: a tuple $(D, b, u, \text{MaxFlowMinCut})$, formed by
- (i) a digraph $D =: (V, A, \varphi)$,
- (ii) a vector $b \in \mathbb{R}^V$ such that $\mathbb{1}^\intercal b = 0$,
- (iii) a function $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$,
- (iv) an algorithm MaxFlowMinCut that takes as input a tuple (D', u', r, s) for the Maximum Flow Problem 11.14 and, provided (MF(D', u', r, s)) has an optimal solution, returns an optimal solution f' for (MF(D', u', r, s)) and a subset $R' \subseteq V(D')$ that is an optimal solution for (14.18) on digraph D' with arc capacities u'.

Output: either a b-transshipment $f \in \mathbb{R}^A_+$, feasible with respect to u, or a subset $S \subseteq V$ such that $b(S) > u\big(\delta^{\text{in}}(S)\big)$; if moreover MaxFlowMinCut ensures that f' is integral if u' is integral, then the transshipment f is integral if b and u are integral.

```
1:  function Transshipment-via-Flows(D, b, u, MaxFlowMinCut)
2:      Let r, s be new vertices, i.e., elements not in V
3:      V' ← V ∪ {r, s}
4:      V+ ← { v ∈ V : b_v > 0}
5:      V− ← { v ∈ V : b_v < 0}
6:      A' ← V− ⊔ A ⊔ V+
7:      for all v ∈ V+ do
8:          φ'((1, v)) ← (r, v)
9:          u'((1, v)) ← −b_v
10:     for all a ∈ A do
11:         φ'((2, a)) ← φ(a)
12:         u'((2, a)) ← u(a)
13:     for all v ∈ V− do
14:         φ'((3, v)) ← (v, s)
15:         u'((3, v)) ← b_v
16:     D' ← (V', A', φ')          ▷ Lines 2 to 16 build an instance (D', u', r, s) as in Exercise 17.3
17:     (f', R') ← MaxFlowMinCut(D', u', r, s)
18:     if u'(δ^out_D'(R')) < b(V+) then
19:         return S := V \ R'
20:     else
21:         return f := ψ(f')      ▷ with ψ defined as in (17.8)
```

17.5 Theorem (Algorithmic Gale's Theorem). Let $D := (V, A, \varphi)$ be a digraph. Let $b \in \mathbb{R}^V$ such that $\mathbb{1}^\intercal b = 0$. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Suppose Algorithm 17.1 is run on input $(D, b, u, \text{Edmonds-Karp})$. Then it returns either

(17.13) (i) a feasible b-transshipment f, which is integral if b and u are integral, or
(ii) a subset $S \subseteq V$ such that $b(S) > u\big(\delta^{\text{in}}(S)\big)$.

Proof. Suppose there is a feasible b-transshipment. Then (17.9)(i) shows that the optimal value of the optimization problem (MF(D', u', r, s)) is $\geq b(V^+)$, and this optimal value is $\leq u'(\delta^{\text{out}}_{D'}(\{r\} \cup V)) = b(V^+)$ by Theorem 13.5 and (17.6). Thus, f' is an rs-flow of value $b(V^+)$, feasible with respect to u'. By Theorem 13.5, the test in **if** in Line 18 must fail, and the return value f is a b-transshipment satisfying the desired conditions by (17.9)(ii).

Next, suppose there is no feasible b-transshipment. If the optimal value of (MF(D', u', r, s)) is $\geq b(V^+)$, then by Theorem 14.10 there is an rs-flow f' in D', feasible with respect to u', of value $\geq b(V^+)$. Thus, (17.9)(ii) shows that $\psi(f')$ is a b-transshipment, feasible with respect to u, a contradiction, since we are supposing no such transshipments exist. Thus, the optimal value of (MF(D', u', r, s)) is $< b(V^+)$, so by Theorem 14.10, the test in **if** in Line 18 passes, and the return value S satisfies the desired conditions by Proposition 17.4. □

17.6 Corollary (Gale's Theorem). Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $b \in \mathbb{R}^V$. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Then precisely one of the following holds:

(17.14) (i) there exists a feasible b-transshipment $f \in \mathbb{R}^A_+$;
(ii) $\mathbb{1}^\intercal b \neq 0$ or there is $S \subseteq V$ such that $b(S) > u\big(\delta^{\text{in}}(S)\big)$.

Moreover, in case (17.14)(i) holds and b and u are integral, there is a feasible b-transshipment that is integral.

Proof. Immediate from Proposition 17.2 and Theorem 17.5. □

17.7 Circulations. Let $D \coloneqq (V, A, \varphi)$ be a digraph. A **circulation**[i] is a 0-transshipment, i.e., conservation of flow (17.2) holds everywhere.

If f is a circulation, then $B_D f = \text{excess}_f = 0$ by (12.5) and (17.1). Hence, by looking at $\mathbb{1}_S^\intercal B_D f = 0$ for a subset $S \subseteq V$, Corollary 12.8 shows that

$$f\big(\delta^{\text{in}}(S)\big) = f\big(\delta^{\text{out}}(S)\big) \qquad \text{for every } S \subseteq V, \text{ if } f \text{ is a circulation.} \tag{17.15}$$

Let $\ell\colon A \to \mathbb{R}_+$ and let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. We say that a circulation $f \in \mathbb{R}^A_+$ is **feasible** (with respect to ℓ and u) if $\ell \leq f \leq u$.

17.8 Proposition. Let $D \coloneqq (V, A, \varphi)$ be a digraph. Let $\ell\colon A \to \mathbb{R}_+$ be a vector and let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. If there is a feasible circulation, then

$$\ell\big(\delta^{\text{in}}(S)\big) \leq u\big(\delta^{\text{out}}(S)\big) \qquad \text{for every } S \subseteq V. \tag{17.16}$$

Proof. Let f be a feasible circulation. Then

$$\begin{aligned} \ell\big(\delta^{\text{in}}(S)\big) &\leq f\big(\delta^{\text{in}}(S)\big) && \text{since } \ell \leq f \\ &= f\big(\delta^{\text{out}}(S)\big) && \text{by (17.15)} \\ &\leq u\big(\delta^{\text{out}}(S)\big) && \text{since } f \leq u. \end{aligned}$$

This completes the proof. □

17.9 Proposition (Circulations and Transshipments). Let $D \coloneqq (V, A, \varphi)$ be a digraph. Let $\ell\colon A \to \mathbb{R}_+$ be a vector and let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Let $f \in \mathbb{R}^A$. Then f is a circulation that is feasible with respect to ℓ and u if only if $f' \coloneqq f - \ell$ is a $(-B_D\ell)$-transshipment that is feasible with respect to $u - \ell$.

Proof. Let $f \in \mathbb{R}^A$, and set

$$f' \coloneqq f - \ell. \tag{17.17}$$

We have

$$\begin{aligned} \text{excess}_{f'} &= B_D f' && \text{by (12.5)} \\ &= B_D(f - \ell) && \text{by (17.17)} \\ &= B_D f - B_D \ell && \text{by (10.17c)} \\ &= \text{excess}_f - B_D \ell && \text{by (12.5).} \end{aligned} \tag{17.18}$$

Necessity: Suppose f is a circulation, feasible with respect to ℓ and u. Thus, $\ell \leq f \leq u$. Clearly, $0 \leq f' \leq u - \ell$. Also, $\text{excess}_{f'} = \text{excess}_f - B_D\ell$ by (17.18), so $\text{excess}_{f'} = -B_D\ell$ follows since f is a circulation.

Sufficiency: Suppose that f' is a $(-B_D\ell)$-transshipment, feasible with respect to $u - \ell$. Thus, $0 \leq f' \leq u - \ell$. Clearly, $\ell \leq f \leq u$. Also, $\text{excess}_f = \text{excess}_{f'} + B_D\ell$ by (17.18), so $\text{excess}_f = 0$. □

17.10 Proposition. Let $D \coloneqq (V, A, \varphi)$ be a digraph. Let $\ell\colon A \to \mathbb{R}_+$ be a vector and let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Set

$$b \coloneqq -B_D\ell \tag{17.19}$$

and

$$u' \coloneqq u - \ell. \tag{17.20}$$

If $S \subseteq V$ is such that $b(S) > u'\big(\delta^{\text{in}}(S)\big)$, then $\ell\big(\delta^{\text{out}}(S)\big) > u\big(\delta^{\text{in}}(S)\big)$.

[i] Dealing with circulations is sometimes "easier" since there are no special vertices in a circulation, as opposed to, e.g., the special vertices r and s in an rs-flow.

Proof. We have

$$\begin{aligned} b(S) &= \mathbb{1}_S^\mathsf{T} b && \text{by (10.12)} \\ &= -\mathbb{1}_S^\mathsf{T} B_D \ell && \text{by (17.19)} \\ &= -(B_D^\mathsf{T} \mathbb{1}_S)^\mathsf{T} \ell && \text{by (10.20)} \\ &= \ell\big(\delta^{\text{out}}(S)\big) - \ell\big(\delta^{\text{in}}(S)\big) && \text{by Corollary 12.8.} \end{aligned} \tag{17.21}$$

Moreover,

$$u'\big(\delta^{\text{in}}(S)\big) = u\big(\delta^{\text{in}}(S)\big) - \ell\big(\delta^{\text{in}}(S)\big) \tag{17.22}$$

by (17.20). The result follows from (17.21) and (17.22). □

Algorithm 17.2 Circulation via Transshipment

Input: a tuple $(D, \ell, u, \textsc{Transshipment})$, formed by
- (i) a digraph $D =: (V, A, \varphi)$,
- (ii) a vector $\ell\colon A \to \mathbb{R}_+$
- (iii) a function $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ such that $\ell \leq u$,
- (iv) an algorithm Transshipment that takes as input a tuple (D', b, u') where $D =: (V', A', \varphi')$ is a digraph, $b \in \mathbb{R}^{V'}$ is a vector such that $\mathbb{1}^\mathsf{T} b = 0$, and $u'\colon A' \to \mathbb{R}_+ \cup \{+\infty\}$ is a function, and returns either a b-transshipment that is feasible with respect to u', or a subset $S \subseteq V'$ such that $b(S) > u'\big(\delta^{\text{in}}_{D'}(S)\big)$.

Output: either a circulation $f \in \mathbb{R}^A$ that is feasible with respect to ℓ and u or a subset $S \subseteq V$ such that $\ell\big(\delta^{\text{out}}(S)\big) > u\big(\delta^{\text{in}}(S)\big)$. If, moreover Transshipment ensures that f' is integral if b and u' are integral, then f is integral if ℓ and u are integral.

```
1: function Circulation-via-Transshipment(D, ℓ, u, Transshipment)
2:     u' := u − ℓ ≥ 0
3:     b := −B_D ℓ
4:     Ret ← Transshipment(D, b, u')
5:     if Ret is a b-transshipment f' that is feasible with respect to u' then
6:         return f := f' + ℓ
7:     else
8:         S := Ret                ▷ S is a subset S ⊆ V such that b(S) > u'(δ^in(S))
9:         return S
```

17.11 Theorem (Algorithmic Hoffman's Theorem). Let $D := (V, A, \varphi)$ be a digraph. Let $\ell\colon A \to \mathbb{R}_+$ be a vector and let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function such that $\ell \leq u$. Suppose that Algorithm 17.2 is run on input $(D, \ell, u, \textsc{Transshipment})$; note that potentially Transshipment might be the algorithm Transshipment-via-Flows$(\cdot,\cdot,\cdot,$ Edmonds-Karp$)$. Then it returns either

(17.23)
- (i) a circulation f that is feasible with respect to ℓ and u, with f integral if ℓ and u are integral, or
- (ii) a subset $S \subseteq V$ such that $\ell\big(\delta^{\text{in}}(S)\big) > u\big(\delta^{\text{out}}(S)\big)$.

Proof. First note that Corollary 12.8 shows that the vector $b = -B_D \ell$ satisfies $\mathbb{1}^\mathsf{T} b = 0$, required by Transshipment.

Suppose there is a circulation f that is feasible with respect to ℓ and u. Then Proposition 17.9 shows that there is a b-transshipment that is feasible with respect to u'. Thus, by Proposition 17.2 the return value Ret in Line 4 is a b-transshipment f' feasible with respect to u'. Hence, the vector returned $f = f' + \ell$ is a feasible circulation satisfying the desired conditions by Proposition 17.9.

Next, suppose there is no circulation f that is feasible with respect to ℓ and u. Then Proposition 17.9 shows that there is no b-transshipment f' that is feasible with respect to u'. Thus, by the hypotheses on Transshipment, the return value Ret in Line 4 is a set $S \subseteq V'$ such that $b(S) > u'\big(\delta^{\text{in}}(S)\big)$. Thus, the subset S returned satisfies the desired conditions by Proposition 17.10. □

17.12 Corollary (Hoffman's Theorem). Let $D \coloneqq (V, A, \varphi)$ be a digraph. Let $\ell\colon A \to \mathbb{R}_+$ be a vector and let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function such that $\ell \leq u$. Then precisely one of the following holds:

(17.24) (i) there exists a feasible circulation f,
(ii) there is $S \subseteq V$ such that $\ell\big(\delta^{\text{in}}(S)\big) > u\big(\delta^{\text{out}}(S)\big)$.

Moreover, in case (17.24)(i) holds and ℓ and u are integral, there is a feasible circulation that is integral.

Proof. Immediate from Proposition 17.8 and Theorem 17.11. □

17.13 Problem (The Minimum-Cost Circulation Problem). The **minimum-cost circulation problem** is, **given**

(i) a digraph $D = (V, A, \varphi)$,
(ii) a cost function $c\colon A \to \mathbb{R}$,
(iii) a vector $\ell\colon A \to \mathbb{R}_+$, and
(iv) a function $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$,

solve the LP

$$(\text{MCC}(D, c, \ell, u)) \qquad \begin{aligned} \text{Minimize} \quad & c^{\mathsf{T}} f \\ \text{subject to} \quad & B_D f = 0, \\ & \ell \leq f \leq u, \\ & f \in \mathbb{R}_+^A. \end{aligned}$$

The **minimum-cost integral circulation problem** is, **given**

(i) a digraph $D = (V, A, \varphi)$,
(ii) a cost function $c\colon A \to \mathbb{R}$,
(iii) an *integral* vector $\ell\colon A \to \mathbb{Z}_+$, and
(iv) an *integral* function $u\colon A \to \mathbb{Z}_+ \cup \{+\infty\}$,

solve the optimization problem

$$(\text{MCIC}(D, c, \ell, u)) \qquad \begin{aligned} \text{Minimize} \quad & c^{\mathsf{T}} f \\ \text{subject to} \quad & B_D f = 0, \\ & \ell \leq f \leq u, \\ & f \in \mathbb{Z}_+^A. \end{aligned}$$

Lecture 18: Decomposition of Flows and Circulations

18.1 Minimal and Maximal Elements. Let $(P, \preccurlyeq)$ be a poset, and let $\prec$ be the strict relation from $\preccurlyeq$, as defined in §16.6. An element $x \in P$ is **minimal** if there is no element $y \in P$ such that $y \prec x$. Similarly, an element $x \in P$ is **maximal** if there is no element $y \in P$ such that $x \prec y$.

It is useful to consider the concepts of minimal and maximal elements of subsets of P. If $X \subseteq P$, then $(X, \preccurlyeq_X)$ is also a poset (verify this!), where $\preccurlyeq_X \coloneqq \preccurlyeq \cap (X \times X)$. We may that $x \in X$ is **maximal in** X if x is a maximal element of the poset $(X, \preccurlyeq_X)$.

18.2 Maximal Paths. Let $D \coloneqq (V, A, \varphi)$ be a digraph. Define a partial order $\preccurlyeq$ on the set of paths of D as follows. For paths P and Q in D, set $P \preccurlyeq Q$ if $V(P) \subseteq V(Q)$ and $A(P) \subseteq A(Q)$; verify that this does define a partial order. With this partial order in context, we may refer to maximal paths, i.e., maximal elements of this poset. Maximal paths are often used in basic graph-theoretic arguments, and finding such beasts (as is often the case with maximal elements) can be made algorithmic[i].

18.3 Exercise. Design a linear-time algorithm that, given a digraph D, outputs a maximal[ii] path in D. Do the same for a maximal path in the set of all paths that start at a prescribed vertex r.

[i] Which is also important for making such arguments algorithmic.
[ii] Maximal in the poset defined in §18.2.

18.4 Supports of Vectors. The **support** of a vector $x \in \mathbb{R}^V$ is

$$\operatorname{supp}(x) \coloneqq \{\, v \in V : x_v \neq 0\}. \tag{18.1}$$

For $\{0,1\}$-valued vectors, the $\operatorname{supp}(\cdot)$ "operation" may be seen as an inverse to the incidence vector $\mathbb{1}$. "operation"[i] from (10.9):

$$\operatorname{supp}(\mathbb{1}_S) = S \qquad \text{for every } S \subseteq V, \tag{18.2a}$$

$$\mathbb{1}_{\operatorname{supp}(x)} = x \qquad \text{for every } x \in \{0,1\}^V. \tag{18.2b}$$

Hence, if $x \in \{0,1\}^V$, then

$$\begin{aligned} \mathbb{1}^{\mathsf{T}} x &= \mathbb{1}^{\mathsf{T}} \mathbb{1}_{\operatorname{supp}(x)} && \text{by (18.2b)} \\ &= |\operatorname{supp}(x)| && \text{by (10.11).} \end{aligned}$$

18.5 Proposition. Let $D \coloneqq (V, A, \varphi)$ be a digraph with $V \neq \varnothing$. If $\delta^{\text{out}}(v) \neq \varnothing$ for every $v \in V$, then D has a cycle.

Proof. Let $P \coloneqq \langle v_0, a_1, v_1, \ldots, a_\ell, v_\ell \rangle$ be a maximal path in D, i.e., a maximal element of the poset defined in §18.2. Since $\delta^{\text{out}}(v_\ell) \neq \varnothing$, there is an arc $b \in \delta^{\text{out}}(v_\ell)$. Thus, there is $x \in V$ such that $\varphi(b) = (v_\ell, x)$. Then $x \in V(P)$, otherwise $P \cdot \langle v_\ell, b, x \rangle$ is a path in D, which contradicts maximality of P. Thus, there is $i \in \{1, \ldots, \ell\}$ such that $x = v_i$, whence $\langle v_i, \ldots, a_\ell, v_\ell \rangle \cdot \langle v_\ell, b, x \rangle$ is a cycle in D. □

18.6 Proposition. Let $D \coloneqq (V, A, \varphi)$ be a digraph. Let $f \in \mathbb{R}^A_+$ be a nonzero circulation. Then there is a cycle C in D such that $A(C) \subseteq \operatorname{supp}(f)$.

Proof. Set $F \coloneqq \operatorname{supp}(f)$ and $V' \coloneqq \{\, v \in V : v \text{ is the head or tail of some arc of } F\}$. Set $D' \coloneqq (V', F, \varphi\restriction_F)$, and note that D' is a digraph by the definition of V'. Moreover, $F \neq \varnothing$ since $f \neq 0$, so $V' \neq \varnothing$. If D' has a loop, we are done, so assume that D' is loopless. We claim that

$$\delta^{\text{out}}_{D'}(v) \neq \varnothing \qquad \text{for each } v \in V'. \tag{18.3}$$

Let $v \in V'$. Then there is $a \in F$ such that v is the head or tail of a. If v is the tail of a, then $a \in \delta^{\text{out}}_{D'}(v)$, thus the condition (18.3) holds for this specific vertex v. Otherwise, v is the head of a, i.e.,

$$a \in \delta^{\text{in}}_D(v). \tag{18.4}$$

Then

$$\begin{aligned} f\big(\delta^{\text{out}}_D(v)\big) = f\big(\delta^{\text{in}}_D(v)\big) &\qquad \text{by (17.2), since } f \text{ is a circulation} \\ \geq f(a) &\qquad \text{by (18.4), since } f \geq 0 \\ > 0 &\qquad \text{since } f \geq 0 \text{ and } a \in F = \operatorname{supp}(f). \end{aligned}$$

Thus, there is $b \in \delta^{\text{out}}_D(v)$ such that $f(b) > 0$, so $b \in \operatorname{supp}(f) = F$ and thus $b \in \delta^{\text{out}}_{D'}(v)$. This completes the proof of (18.3).

Equation (18.3) shows that the hypotheses of Proposition 18.5 applied to D' are met, and the result now follows from Proposition 18.5. □

18.7 Theorem (Decomposition of Circulations). Let $D \coloneqq (V, A, \varphi)$ be a digraph. If $f \in \mathbb{R}^A_+$ is a circulation, then there are a set $\mathcal{C}$ of cycles of $(V, \operatorname{supp}(f), \varphi\restriction_{\operatorname{supp}(f)})$ and a vector $y \in \mathbb{R}^{\mathcal{C}}_+$ such that $f = \sum_{C \in \mathcal{C}} y_C \mathbb{1}_C$ and $|\mathcal{C}| \leq |\operatorname{supp}(f)|$. If, moreover, f is integral, then the previous statement holds with the additional conclusion that y is integral.

Proof. The proof is by induction on $|\operatorname{supp}(f)|$. If $|\operatorname{supp}(f)| = 0$, then $\operatorname{supp}(f) = \varnothing$ so $f = 0$ and the result follows by taking $\mathcal{C} \coloneqq \varnothing$. Suppose that $\operatorname{supp}(f) \neq \varnothing$. By Proposition 18.6, there is a cycle C of D such that

$$A(C) \subseteq \operatorname{supp}(f). \tag{18.5}$$

Set

$$\mu \coloneqq \min_{a \in A(C)} f(a), \tag{18.6}$$

[i] Typography is quite inconvenient here; note the little dot · as a subscript.

and note that $\mu > 0$ by (18.5), since $f \geq 0$. By construction,

$$f \geq \mu \mathbb{1}_C. \tag{18.7}$$

Indeed, let $a \in A$. If $a \notin A(C)$, then $f(a) \geq 0 = \mu \mathbb{1}_C(a)$. If $a \in A(C)$, then $\mu \leq f(a)$ by (13.17) so $f(a) \geq \mu = \mu \mathbb{1}_C(a)$. This proves (18.7), whence

$$f' \coloneqq f - \mu \mathbb{1}_C \geq 0. \tag{18.8}$$

Thus,

$$\operatorname{supp}(f') \subseteq \operatorname{supp}(f). \tag{18.9}$$

By Theorem 1.5, there is $b \in A(C)$ such that $\mu = f(b)$, so

$$\begin{aligned} f'(b) &= 0 && \text{by (18.8)} \\ &< f(b) && \text{by (18.5), since } b \in A(C) \text{ and } f \geq 0. \end{aligned}$$

This proves that the inclusion in (18.9) is strict, so

$$|\operatorname{supp}(f')| \leq |\operatorname{supp}(f)| - 1. \tag{18.10}$$

Finally,

$$\begin{aligned} B_D f' &= B_D f - \mu B_D \mathbb{1}_C && \text{by (18.8) and (10.17c)} \\ &= -\mu B_D \mathbb{1}_C && \text{since } f \text{ is a circulation} \\ &= 0 && \text{by (11.14)}, \end{aligned}$$

i.e., f' is a circulation. By (18.10) and induction hypothesis, there are a set $\mathcal{C}'$ of cycles of the digraph $(V, \operatorname{supp}(f'), \varphi\restriction_{\operatorname{supp}(f')})$ and a vector $y' \in \mathbb{R}_+^{\mathcal{C}'}$ such that

$$f' = \sum_{B \in \mathcal{C}'} y'_B \mathbb{1}_B, \tag{18.11a}$$

$$|\mathcal{C}'| \leq |\operatorname{supp}(f')|, \tag{18.11b}$$

$$\text{and with } y' \text{ integral if } f' \text{ is integral.} \tag{18.11c}$$

Set

$$\mathcal{C} \coloneqq \mathcal{C}' \cup \{C\} \tag{18.12}$$

and extend[1] $y' \in \mathbb{R}_+^{\mathcal{C}'}$ to $y \in \mathbb{R}_+^{\mathcal{C}}$ by setting $y_C \coloneqq \mu$, i.e.,

$$y\restriction_{\mathcal{C}'} = y', \tag{18.13a}$$

$$y_C = \mu. \tag{18.13b}$$

Then

$$\begin{aligned} f &= \mu \mathbb{1}_C + f' && \text{by (18.8)} \\ &= \mu \mathbb{1}_C + \sum_{B \in \mathcal{C}'} y'_B \mathbb{1}_B && \text{by (18.11a)} \\ &= y_C \mathbb{1}_C + \sum_{B \in \mathcal{C}'} y_B \mathbb{1}_B && \text{by (18.13)} \\ &= \sum_{B \in \mathcal{C}} y_B \mathbb{1}_B && \text{by (18.12)}. \end{aligned}$$

Moreover[i],

$$\begin{aligned} |\mathcal{C}| &\leq |\mathcal{C}'| + 1 && \text{by (18.12)} \\ &\leq |\operatorname{supp}(f')| + 1 && \text{by (18.11b)} \\ &\leq |\operatorname{supp}(f)| && \text{by (18.10)}. \end{aligned}$$

[1] If f is a function with domain X and g is a function with domain Y, we say that f is an **extension** of g if $Y \subseteq X$ and $g = f\restriction_Y$. In particular, if f is an extension of a function g already defined, then it suffices to define $f(x)$ for each $x \in X \setminus Y$.

[i] The first inequality is actually an equation, since the above shows that $C \notin \mathcal{C}'$. However, we do not need this fact.

Finally, if f is integral, then μ is an integer by (18.6), and so f' is integral by (18.8); then, (18.13) and (18.11c) show that y is integral. □

18.8 The True Nature of Circulations. Let $D \coloneqq (V, A, \varphi)$ be a digraph. From (11.14), it was already known that, if C is a cycle in D, then $\mathbb{1}_C \in \mathbb{R}^A_+$ is a circulation in D. In fact, if $\mathcal{C}$ is a set of cycles in D and $y \in \mathbb{R}^{\mathcal{C}}_+$ is a vector, then

$$\begin{aligned} B_D\Big(\sum_{C\in\mathcal{C}} y_C \mathbb{1}_C\Big) &= \Big(\sum_{C\in\mathcal{C}} y_C B_D \mathbb{1}_C\Big) && \text{by (10.17c)} \\ &= 0 && \text{by (11.14).} \end{aligned}$$

Thus, every linear combination of such vectors with nonnegative coefficients is also a circulation in $\mathbb{R}^A_+$. Linear combinations with nonnegative coefficients are called *conic* combinations, and we will use this term from now on. Thus, every conic combination of (incidence vectors of) cycles in D is a circulation in D.

Theorem 18.7 provides a strong converse to this fact: every circulation in D is a conic combination of (incidence vectors of) cycles in D, along with some extra properties regarding support size and integrality.

18.9 Flows and Circulations. Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Let b be a *new* arc, i.e., an element not in A, and set $D^+ \coloneqq (V, A^+, \varphi^+)$, where $A^+ \coloneqq A \cup \{b\}$ and φ^+ is the extension of φ to A^+ defined by setting $\varphi^+(b) \coloneqq (s, r)$.

Given $f \in \mathbb{R}^A$, define f^+ as the extension of f to A^+ defined by setting $f^+(b) \coloneqq \operatorname{excess}_f(s)$. Verify that

$$B_{D^+} f^+ = B_D f + \big(\operatorname{excess}_f(s)\big)(e_r - e_s). \tag{18.14}$$

Thus, if $\mathcal{F}_{rs}$ denotes the set of rs-flows in D and $\mathcal{C}^+$ denotes the set of circulations in D^+, then the map

$$\psi\colon f \in \mathcal{F}_{rs} \mapsto f^+ \in \mathcal{C}^+ \tag{18.15}$$

is a bijection.

18.10 Exercise (Decomposition of Flows). Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Prove[1] that, if $f \in \mathbb{R}^A_+$ is an rs-flow, then there are

(i) a set $\mathcal{C}$ of cycles of $(V, \operatorname{supp}(f), \varphi\restriction_{\operatorname{supp}(f)})$ and a vector $y \in \mathbb{R}^{\mathcal{C}}_+$, and
(ii) a set $\mathcal{P}$ of rs-paths of $(V, \operatorname{supp}(f), \varphi\restriction_{\operatorname{supp}(f)})$ and a vector $x \in \mathbb{R}^{\mathcal{P}}_+$,

such that

$$\begin{aligned} f &= \sum_{C\in\mathcal{C}} y_C \mathbb{1}_C + \sum_{P\in\mathcal{P}} x_P \mathbb{1}_P, \\ \mathbb{1}^\mathsf{T} x &= \operatorname{value}(f), \\ |\mathcal{C}| + |\mathcal{P}| &\leq |\operatorname{supp}(f)|. \end{aligned}$$

Furthermore show that, if f is integral, then one can take y and x integral.

18.11 The True Nature of Flows. Let $D \coloneqq (V, A, \varphi)$ be a digraph, and let $r, s \in V$ be distinct. Continuing the discussion from §18.8, it was already known that, if P is an rs-path in D, then $\mathbb{1}_P \in \mathbb{R}^A_+$ is an rs-flow in D by (11.14). Again from (11.14), any conic combination of (incidence vectors of) cycles plus any conic combination of (incidence vectors of) rs-paths yields an rs-flow.

Exercise 18.10 describes a strong converse to this fact, again with extra properties.

18.12 Reducing Max-Flow to Min-Cost Circulations. Restore the context of §18.9. Let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function. Define $\ell \coloneqq 0 \in \mathbb{R}^{A^+}_+$, and define the extension u^+ of u to A^+ by setting $u^+(b) \coloneqq +\infty$. Finally, define $c \coloneqq -e_b \in \mathbb{R}^{A^+}$, so that (D^+, c, ℓ, u^+) determines an instance of the Minimum-Cost Circulation Problem 17.13.

By §18.9, if f is a feasible rs-flow in D, then $\psi(f)$ is a feasible solution for (MCC(D^+, c, ℓ, u^+)) with objective value $-e_b^\mathsf{T} f^+ = -\operatorname{value}(f)$. Conversely, if g is a feasible solution for (MCC(D^+, c, ℓ, u^+)) with objective value $-\mu$, then $\psi^{-1}(g)$ is a feasible rs-flow of value μ. Both statements may be adapted to their integral versions.

Hence, any efficient algorithm for the Minimum-Cost Circulation Problem 17.13 provides an efficient algorithm for the Maximum Flow Problem 11.14, i.e., the latter problem is at least as easy as the former.

[1] Naturally, it will be easier to do this if you rely on Theorem 18.7 as a black-box, along with the construction from §18.9.

18.13 Problem (The Minimum-Cost Flow Problem). The **minimum-cost flow problem** is, **given**

(i) a digraph $D = (V, A, \varphi)$,
(ii) a cost function $c\colon A \to \mathbb{R}$,
(iii) a function $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$,
(iv) distinct vertices $r, s \in V$,
(v) a nonnegative number $\beta \in \mathbb{R}_+$,

solve the LP

$$\begin{aligned} \text{Minimize} \quad & c^\mathsf{T} f \\ \text{subject to} \quad & B_D f = \beta(e_s - e_r), \\ & f \leq u, \\ & f \in \mathbb{R}_+^A. \end{aligned} \tag{18.16}$$

The **minimum-cost integral flow problem** is, **given**

(i) a digraph $D = (V, A, \varphi)$,
(ii) a cost function $c\colon A \to \mathbb{R}$,
(iii) an *integral* function $u\colon A \to \mathbb{Z}_+ \cup \{+\infty\}$,
(iv) a nonnegative *integer* $\beta \in \mathbb{Z}_+$,

solve the optimization problem

$$\begin{aligned} \text{Minimize} \quad & c^\mathsf{T} f \\ \text{subject to} \quad & B_D f = \beta(e_s - e_r), \\ & f \leq u, \\ & f \in \mathbb{Z}_+^A. \end{aligned} \tag{18.17}$$

Note that, obtaining a *feasible* solution to either of these problems reduces to solving a Maximum (Integer) Flow Problem 11.14. In this sense, the Minimum Cost Flow Problem is at least as hard as the Maximum Flow Problem.

18.14 Exercise (Reducing Shortest Paths to Min-Cost Flow). Show how the LP (11.15) fits as an instance of (18.16). Does Exercise 18.10 provide any insight into why the optimal value of (11.15) is equal to the minimum cost of an rs-path?

18.15 Exercise (Reducing Min-Cost Flows to Min-Cost Circulations). Design an efficient algorithm that, given any instance (D, c, u, β) for the Minimum-Cost Flow Problem 18.13, builds an instance of the Minimum-Cost Circulation Problem 17.13 so that the corresponding optimization problems (18.16) and (MCC(D, c, ℓ, u)) are equivalent. Naturally, both homomorphisms provided should be efficient algorithms.

18.16 Problem (The Minimum-Cost Transshipment Problem). The **minimum-cost transshipment problem** is, **given**

(i) a digraph $D = (V, A, \varphi)$,
(ii) a cost function $c\colon A \to \mathbb{R}$,
(iii) a function $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$,
(iv) a vector $b \in \mathbb{R}^V$ such that $\mathbb{1}^\mathsf{T} b = 0$,

solve the LP

$$\begin{aligned} \text{Minimize} \quad & c^\mathsf{T} f \\ \text{subject to} \quad & B_D f = b, \\ & f \leq u, \\ & f \in \mathbb{R}_+^A. \end{aligned} \tag{18.18}$$

The **minimum-cost integral transshipment problem** is, **given**

(i) a digraph $D = (V, A, \varphi)$,
(ii) a cost function $c\colon A \to \mathbb{R}$,
(iii) an *integral* function $u\colon A \to \mathbb{Z}_+ \cup \{+\infty\}$,
(iv) an *integral* vector $b \in \mathbb{Z}^V$ such that $\mathbb{1}^\mathsf{T} b = 0$,

solve the optimization problem

$$\begin{aligned} &\text{Minimize} && c^{\mathsf{T}} f \\ &\text{subject to} && B_D f = b, \\ &&& f \leq u, \\ &&& f \in \mathbb{Z}_+^A. \end{aligned} \tag{18.19}$$

18.17 Exercise (Reductions Between Min-Cost Circulation and Min-Cost Transshipment). Provide reductions between these two problems, in both directions, in the same sense as Exercise 18.15.

Lecture 19: Primal, Dual, and Primal-Dual Algorithms

19.1 Problem (The Minimum-Cost Perfect Matching Problem). The **minimum-cost perfect matching problem** is, **given**

(i) a graph $G = (V, E, \varphi)$,
(ii) a cost function $c\colon E \to \mathbb{R}$,

solve the optimization problem

$$\begin{aligned} &\text{Minimize} && c(M) \\ &\text{subject to} && M \text{ is a perfect matching in } G. \end{aligned} \tag{19.1}$$

The **bipartite minimum-cost perfect matching problem** is, **given**

(i) a graph $G = (V, E, \varphi)$,
(ii) sets $U, W \subseteq V$ such that G is (U, W)-bipartite and $|U| = |W|$, and
(iii) a cost function $c\colon E \to \mathbb{R}$,

solve the optimization problem (19.1). Note that the condition $|U| = |W|$ is necessary for the existence of a perfect matching in G.

19.2 Exercise (Reduction to Min-Cost Flow in Bipartite Case). Let $G = (V, E, \varphi)$ be a (U, W)-bipartite graph such that $|U| = |W|$, and let $c\colon E \to \mathbb{R}$ be a vector. Define digraph D and function $\bar{u}\colon A(D) \to \mathbb{R}_+ \cup \{+\infty\}$ as in §12.3. Prove that (19.1) is isomorphic to (18.17) on digraph D, capacities $\bar{u}$, and target flow value $\beta := |U| = |W|$, with isomorphisms as described in §12.6 and §13.1.

19.3 Integer and LP Relaxations. Let $G = (V, E, \varphi)$ be a graph, and let $c\colon E \to \mathbb{R}$ be a vector, and define the optimization problem[1] (which is an IP)

$$\begin{aligned} &\text{Minimize} && c^{\mathsf{T}} x && \text{(19.2a)} \\ &\text{subject to} && B_G x = \mathbb{1}, && \text{(19.2b)} \\ &&& x \in \mathbb{R}_+^E, && \text{(19.2c)} \\ &&& x \in \mathbb{Z}^E. && \text{(19.2d)} \end{aligned}$$

Note that (19.2b) encodes the degree constraints $x\big(\delta(v)\big) = 1$ for each $v \in V$; see §9.5 and §10.16. Prove[i] (exercise!) that

$$\text{the map } M \subseteq E \mapsto \mathbb{1}_M \in \mathbb{R}^E \text{ is an isomorphism from (19.1) to (19.2).} \tag{19.3}$$

The LP relaxation of (19.2) is obtained by dropping the integrality constraint (19.2d):

$$\begin{aligned} &\text{Minimize} && c^{\mathsf{T}} x \\ &\text{subject to} && B_G x = \mathbb{1}, \\ &&& x \in \mathbb{R}_+^E, \end{aligned} \tag{19.4}$$

[1] Compare (19.2) to (11.7)

[i] Rely on Exercise 10.17 to relate the matrix equation from (19.2) to the degree constraints from §9.5.

and its LP dual is

$$\begin{aligned} &\text{Maximize} && \mathbb{1}^{\mathsf{T}} y \\ &\text{subject to} && y \in \mathbb{R}^V, \\ & && B_G^{\mathsf{T}} y \leq c. \end{aligned}$$

By (10.28a), the latter can be rewritten as

$$\begin{aligned} &\text{Maximize} && y(V) && \text{(19.5a)} \\ &\text{subject to} && y \in \mathbb{R}^V, && \text{(19.5b)} \\ & && y\big(\varphi(e)\big) \leq c_e \quad \text{for each } e \in E. && \text{(19.5c)} \end{aligned}$$

The inequality constraint (19.5c) can be rewritten as

$$y_u + y_w \leq c_e \quad \text{for each } e \in E, \text{ where } uw := \varphi(e).$$

Let $M \subseteq E$ be a perfect matching in G. Then $\mathbb{1}_M$ is feasible in (19.2), and thus also in (19.4). Let y be a feasible solution for (19.5). By LP Weak Duality (see Theorem 11.2), we have

$$c(M) = c^{\mathsf{T}} \mathbb{1}_M \geq y(V). \tag{19.6}$$

Also, by (11.4),

(19.7) equality holds in (19.6) if and only if $y\big(\varphi(e)\big) = c_e$ whenever $e \in M$;

see also the discussion about complementary slackness in §11.4. Achieving equality in (19.6) ensures that M is an optimal solution for (19.1).

This motivates the following approach[i] for designing an efficient algorithm for (19.1): search for a perfect matching $M \subseteq E$ and a vector $y \in \mathbb{R}^V$ that satisfy the conditions from (19.7).

19.4 Exercise. Let $G = (V, E, \varphi)$ be a graph, and let $c\colon E \to \mathbb{R}$ be a vector. Prove that the vector $y \in \mathbb{R}^V$ defined by

$$y := \tfrac{1}{2}[E \neq \varnothing] \min\{\, c_e : e \in E\} \cdot \mathbb{1} \tag{19.8}$$

is feasible in (19.5).

19.5 Exercise. Let $G = (V, E, \varphi)$ be a graph, and let $c\colon E \to \mathbb{R}$ be a vector. Prove that, if there is a vector $d \in \mathbb{R}^V$ such that $\mathbb{1}^{\mathsf{T}} d > 0$ and $B_G^{\mathsf{T}} d \leq 0$, then (19.5) is unbounded, and therefore G has no perfect matching.

19.6 Primal Algorithms and Dual Algorithms. An *informal* terminology used to describe how some optimization algorithms function is the following. Suppose the algorithm somehow solves an LP, which is potentially an LP relaxation of an integer program. Such LP is called the primal, and in designing the algorithm, one may refer to the dual LP, and to "dual solutions", i.e., feasible solutions of the dual LP.

A most natural form of designing algorithms is to build iterates, each of which is a feasible solution of the primal LP, and each iteration "improves" this primal (feasible) solution in some way. This is the approach, for instance, of Algorithms 8.1 and 14.1. At the end of these algorithms, the current *primal* solution attains optimality, and one in fact builds, **at the very end**, a dual optimal solution that *certifies* optimality of the *final* primal solution. Such algorithms are called *primal algorithms*. Potentially, the optimal dual solution found at the end is *the very first dual feasible solution found*!

A less natural though powerful approach is to keep, at each iteration a *dual solution*, which is improved at each iteration. Typically, at each iteration, one tries to build a primal (feasible) solution that satisfies the complementary slackness conditions (see §11.4) with the *current dual solution*. If one is found, optimality is obtained; otherwise, the algorithm somehow may use this fact (that no primal solution as desired was found) to find an improvement (in the form of a search direction, as in §11.7) to the current dual solution. Such algorithms are called *dual algorithms*. Potentially, the optimal primal solution found at the end is *the very first primal feasible solution found*!

[i] At this point, this so-called approach is just *wishful thinking*. Indeed, if the LP relaxation (19.4) does not have an integral optimal solution, there is no chance that equality can be attained as in (19.6), and the approach is doomed. With the benefit of hindsight, "on the shoulders of giants" and whatnot, this approach is very powerful and it follows the fundamental approach from §11.6. Even if the approach fails, one often gains insight into the problem that enables adaptations or brand new methods of solution. In fact, this course is all about the most favorable problems where the fundamental approach works, sometimes with (hopefully slight) adaptations.

19.7 A Dual Approach for Bipartite Min-Cost Perfect Matching. Let $G = (V, E, \varphi)$ be a (U, W)-bipartite graph such that $|U| = |W|$, and let $c\colon E \to \mathbb{R}$ be a vector. Let y be feasible for (19.5). Define the set of *equality edges* (of y)

$$E^{=}(y) \coloneqq \big\{ e \in E : y\big(\varphi(e)\big) = c_e \big\}, \tag{19.9}$$

as well as the *equality graph*

$$G^{=}(y) \coloneqq (V, E^{=}(y), \varphi\restriction_{E^{=}(y)}) \tag{19.10}$$

formed only by the equality edges. By (19.6) and (19.7),

(19.11) if M is a perfect matching in $G^{=}(y)$, then M is optimal for (19.1).

Let $K \subseteq V$ be a vertex cover of $G^{=}(y)$. We will argue[i] that

$$d \coloneqq \mathbb{1}_U - \mathbb{1}_K \in \{0, \pm 1\}^V. \tag{19.12}$$

is a good search direction (in the same sense as discussed in §13.9) starting from y, if $|K| < |U|$. Note that

$$\begin{aligned} d\big(\varphi(e)\big) &= \mathbb{1}_{\varphi(e)}^{\mathsf{T}} d && \text{by (10.12)} \\ &= \mathbb{1}_{\varphi(e)}^{\mathsf{T}} \big(\mathbb{1}_U - \mathbb{1}_K\big) && \text{by (19.12)} \\ &= \mathbb{1}_{\varphi(e)}^{\mathsf{T}} \mathbb{1}_U - \mathbb{1}_{\varphi(e)}^{\mathsf{T}} \mathbb{1}_K && \text{by (10.17c)} \\ &= |\varphi(e) \cap U| - |\varphi(e) \cap K| && \text{by (10.11)}, \end{aligned} \qquad \text{for each } e \in E. \tag{19.13}$$

Define

$$\alpha \coloneqq \inf\left\{ \frac{c_e - y\big(\varphi(e)\big)}{d\big(\varphi(e)\big)} : e \in E \setminus E^{=}(y),\, d\big(\varphi(e)\big) > 0 \right\} \in (0, +\infty]. \tag{19.14}$$

We will prove that,

(19.15) for every $\lambda \in \mathbb{R}$ such that $0 \leq \lambda \leq \alpha$, the vector $y + \lambda d$ is feasible in (19.5) with objective value $y(V) + \lambda\big(|U| - |K|\big)$.

Performing this change in y when $|K| < |U|$ is referred to as a dual update/improvement step.

Proof. Let us prove that

$$d\big(\varphi(e)\big) \leq 0 \qquad \text{for each } e \in E^{=}(y). \tag{19.16}$$

Let $e \in E^{=}(y)$. Since K is a vertex cover of $G^{=}(y)$, we have

$$|\varphi(e) \cap K| \geq 1, \tag{19.17}$$

whence

$$\begin{aligned} d\big(\varphi(e)\big) &= |\varphi(e) \cap U| - |\varphi(e) \cap K| && \text{by (19.13)} \\ &\leq |\varphi(e) \cap U| - 1 && \text{by (19.17)} \\ &\leq 0 && \text{since } G \text{ is } (U, W)\text{-bipartite.} \end{aligned}$$

This proves (19.16).

Let $\lambda \in \mathbb{R}$ such that $0 \leq \lambda \leq \alpha$, and define

$$y' \coloneqq y + \lambda d. \tag{19.18}$$

Let us verify that

$$y' \text{ is feasible for (19.5).} \tag{19.19}$$

Let $e \in E$. If

$$d\big(\varphi(e)\big) \leq 0, \tag{19.20}$$

which in particular occurs if $e \in E^{=}(y)$ by (19.16), one has

$$\begin{aligned} y'\big(\varphi(e)\big) &= y\big(\varphi(e)\big) + \lambda d\big(\varphi(e)\big) && \text{by (19.18)} \\ &\leq y\big(\varphi(e)\big) && \text{by (19.20), since } \lambda \geq 0 \end{aligned}$$

[i] At this point, this is not a precise mathematical statement.

$$\leq c_e \qquad \text{since } y \text{ is feasible for (19.5).}$$

Suppose next that

$$d\big(\varphi(e)\big) > 0. \tag{19.21}$$

Then $e \notin E^{=}(y)$ by (19.16), so

$$\alpha d\big(\varphi(e)\big) \leq c_e - y\big(\varphi(e)\big) \tag{19.22}$$

by (19.14). Since $0 \leq \lambda \leq \alpha$, by (19.21) we have

$$\lambda d\big(\varphi(e)\big) \leq \alpha d\big(\varphi(e)\big), \tag{19.23}$$

and by combining (19.22) and (19.23) we get $y'\big(\varphi(e)\big) \leq c_e$. This concludes the proof of (19.19).

To compute the objective value of y' in (19.5), note that

$$\begin{aligned} \mathbb{1}^\mathsf{T} d &= \mathbb{1}^\mathsf{T}\big(\mathbb{1}_U - \mathbb{1}_K\big) && \text{by (19.12)} \\ &= \mathbb{1}^\mathsf{T}\mathbb{1}_U - \mathbb{1}^\mathsf{T}\mathbb{1}_K && \text{by (10.17c)} \\ &= |U| - |K| && \text{by (10.11),} \end{aligned}$$

so

$$\mathbb{1}^\mathsf{T} y' = \mathbb{1}^\mathsf{T} y + \lambda \mathbb{1}^\mathsf{T} d = y(V) + \lambda\big(|U| - |K|\big). \qquad \square$$

19.8 A Dual Algorithm. The approach from §19.7 is implemented as Algorithm 19.1. It follows from (19.14) and (19.15) that each iterate y_{t+1} has higher objective value (in (19.5)) than its predecessor y_t. However, as in Algorithm 14.1, it is not clear whether Algorithm 19.1 terminates. One way around this is to somehow ensure "combinatorial progress", as discussed in §15.8; see also Corollaries 16.10 and 16.11. This is what we proceed to do.

Algorithm 19.1 A Dual Algorithm for Bipartite Min-Cost Perfect Matching

Input: a tuple $(G, U, W, c, \textsc{BipMaxMatching})$, formed by
- (i) a bipartite graph $G = (V, E, \varphi)$,
- (ii) subsets $U, W \subseteq V$ such that G is (U, W)-bipartite and $|U| = |W|$,
- (iii) a vector $c\colon E \to \mathbb{R}$,
- (iv) an algorithm BipMaxMatching that takes as input a tuple (G', U', W') for the Bipartite Matching Problem 7.11 and returns an ordered pair (M, K), where M is a maximum matching of G' and K is a minimum vertex cover of G'.

Output: if it terminates, either one of
- (i) a perfect matching M of G and a feasible solution y for (19.5) such that $c(M) = y(V)$, or
- (ii) a vector $d \in \mathbb{R}^V$ such that $\mathbb{1}^\mathsf{T} d > 0$ and $B_G^\mathsf{T} d \leq 0$, as in Exercise 19.5.

function Dual-Algorithm-Bip-Min-Cost-Perfect-Matching$(G, U, W, c, \textsc{BipMaxMatching})$
  Define y_0 as in (19.8)
  for $t \leftarrow 0$ **to** ∞ **do**
    $E_t \leftarrow E^{=}(y_t)$, defined as in (19.9)
    $G_t \leftarrow (V, E_t, \varphi\restriction_{E_t})$
    $(M_t, K_t) \leftarrow \textsc{BipMaxMatching}(G_t, U, W)$
    $d_t \leftarrow \mathbb{1}_U - \mathbb{1}_{K_t}$
    if $|M_t| = |U|$ **then**
      return M_t and y_t
    else if $\{\, e \in E : d_t\big(\varphi(e)\big) > 0 \,\} = \varnothing$ **then**
      return d_t
    else
      $\lambda_t \leftarrow \min\left\{ \dfrac{c_e - y_t\big(\varphi(e)\big)}{d_t\big(\varphi(e)\big)} : e \in E \setminus E^{=}(y_t),\ d_t\big(\varphi(e)\big) > 0 \right\}$
      $y_{t+1} \leftarrow y_t + \lambda_t d_t$

19.9 Primal-Dual Algorithms. One may consider the drawback in Algorithm 19.1 that, in each new iteration, a maximum matching is searched *from scratch*, even though a matching from a previous iteration may still be a matching in the current equality graph (potentially with some edges removed?). In this sense, the use of the black-box algorithm BipMaxMatching may seen wasteful[i]. It might be useful to keep the algorithm *aware* of a current matching at all times, *to help bound/drive the (combinatorial) progress of the algorithm.*

Even though a maximum matching that is not perfect is not a feasible solution in the primal (19.4), it can be considered a "partially" feasible solution. In this sense, the algorithm that we shall build next, which augments Algorithm 19.1 by keeping a matching at each iteration, is called a *primal-dual algorithm*[1ii].

19.10 The Hungarian Method. Algorithm 19.2 is obtained from Algorithm 19.1 by *grafting*[iii] Kőnig's Algorithm 8.1, and keeping not only the dual solution y_t at each iteration, but also a matching M_t in the equality graph $G^{=}(y_t) =: G_t$. At each iteration, one seeks for an M_t-augmenting path in the orientation $D(G_t, U, W, M_t)$, defined as in Exercise 7.14. If one is found, M_t is augmented to a larger matching. Such improvement is monotonic: we *never* decrease the size of the current matching M_t.

If no augmenting path is found, we are back at the same situation in Algorithm 19.1, and a dual improvement step occurs, as in (19.15). We will later prove that, in this case, the set R_t of reachable vertices from M_t-free vertices in U *strictly increases*. This is a form of combinatorial progress: either the matching M_t increases in size[iv], and this can occur at most $O(n)$ times, or else M_t remains the same size, *but* R_t increases[v], and this can happen at most $O(n)$ times as well. This will take us back to lexicographic progress similar to what occurs in Corollaries 16.10 and 16.11.

One *key takeaway* is the certificates of optimality are not just "for show"[vi], but also they are useful for (efficient!) *algorithm design*.

Lecture 20: The Hungarian Method

20.1 Let $G = (V, E, \varphi)$ be a (U, W)-bipartite graph such that $|U| = |W|$, and let $c\colon E \to \mathbb{R}$ be a vector. We start with some locally[vii] scoped definitions and notation.

Let (M, y) be an ordered pair such that

(20.1) (i) y is a feasible solution for (19.5), and
(ii) M is a matching in the graph $G^{=}(y) = (V, E^{=}(y), \varphi\restriction_{E^{=}(y)})$ defined as in (19.10) and (19.9).

Define

$$D(M, y) \coloneqq D\big(G^{=}(y), U, W, M\big) \tag{20.2}$$

as in Exercise 7.14,

$$R(M, y) \coloneqq \big\{\, v \in V : \text{there is } u \in U \setminus V_M \text{ such that } u \rightsquigarrow_{D(M,y)} v \big\} \subseteq V, \tag{20.3}$$

and[viii]

$$L(M, y) \coloneqq \Big(|M|, \big|R(M, y)\big|\Big) \in \mathbb{N} \times \mathbb{N}. \tag{20.4}$$

---

[1] Contrast to §19.6.

[i] However, note that this does not affect polynomiality of the running time.
[ii] Readers interested in further discussion about primal, dual, and primal-dual algorithms are referred to [9, Section 18.5b].
[iii] This is in stark contrast to the approach we take throughout the course, of using other results/algorithms as black-boxes. Note, however, our approach: we first verified whether we could get away with using Kőnig's Algorithm 8.1 as a black-box. Only after we realize that that might not be enough, we pulled our sleeves and decided to muck our hands with the bowels/innards of Kőnig's Algorithm 8.1. The takeaway message is that getting our hands full of gunk is sometimes necessary, though it should be avoided whenever possible. Also, do not spill the gunk onto others if you have the option.
[iv] Read this as $|M_{t+1}| > |M_t|$.
[v] Read this as $|R_{t+1}| > |R_t|$.
[vi] Though they do make a hell of a show!
[vii] Meaning, the scope of these abbreviations lasts only as long as this lecture.
[viii] The function in (20.4) has the same role as (16.39) in the analysis of Algorithm 19.2.

Algorithm 19.2 The Hungarian Method

Input: a tuple (G, U, W, c), formed by
(i) a bipartite graph $G = (V, E, \varphi)$,
(ii) subsets $U, W \subseteq V$ such that G is (U, W)-bipartite and $|U| = |W|$, and
(iii) a vector $c\colon E \to \mathbb{R}$.
Output: either one of
(i) a perfect matching M of G and a feasible solution y for (19.5) such that $c(M) = y(V)$, or
(ii) a vector $d \in \mathbb{R}^V$ such that $\mathbb{1}^\intercal d > 0$ and $B_G^\intercal d \leq 0$, as in Exercise 19.5.

function HUNGARIAN-METHOD(G, U, W, c)
  $M_0 \leftarrow \varnothing$
  Define y_0 as in (19.8)
  for $t \leftarrow 0$ **to** ∞ **do**
    if $|M_t| = |U|$ **then**
      return (M_t, y_t)
    $E_t \leftarrow E^=(y_t)$, defined as in (19.9)
    $G_t \leftarrow (V, E_t, \varphi\restriction_{E_t})$
    $D_t \leftarrow D(G_t, U, W, M_t)$, defined as in Exercise 7.14 ▷ Graft Algorithm 8.1 onto Algorithm 19.1
    $R_t \leftarrow \{ v \in V :$ there is $u \in U \setminus V_{M_t}$ such that $u \rightsquigarrow_{D_t} v\}$
    if $R_t \cap (W \setminus V_{M_t}) \neq \varnothing$ **then**
      Let $w_t \in R_t \cap (W \setminus V_{M_t})$
      Let $u_t \in U \setminus V_{M_t}$ and let P_t be a (u_t, w_t)-path in D_t
      $M_{t+1} \leftarrow M_t \,\Delta\, E(P_t)$ ▷ M_{t+1} is a matching in G_t
      $y_{t+1} \leftarrow y_t$
    else
      $K_t \leftarrow \big(U \setminus R_t\big) \cup \big(W \cap R_t\big)$
      $d_t \leftarrow \mathbb{1}_U - \mathbb{1}_{K_t}$
      if $\{ e \in E : d_t\big(\varphi(e)\big) > 0\} = \varnothing$ **then**
        return d_t
      else
        $\lambda_t \leftarrow \min\left\{ \dfrac{c_e - y_t\big(\varphi(e)\big)}{d_t\big(\varphi(e)\big)} : e \in E \setminus E^=(y_t),\, d_t\big(\varphi(e)\big) > 0 \right\}$
        $y_{t+1} \leftarrow y_t + \lambda_t d_t$
        $M_{t+1} \leftarrow M_t$

20.2 Proposition. Let $G = (V, E, \varphi)$ be a (U, W)-bipartite graph such that $|U| = |W|$, and let $c\colon E \to \mathbb{R}$ be a vector. Let an ordered pair (M, y) satisfy (20.1), and suppose that

$$R(M, y) \cap (W \setminus V_M) = \varnothing. \tag{20.5}$$

Set

$$K \coloneqq \big(U \setminus R(M, y)\big) \cup \big(W \cap R(M, y)\big), \tag{20.6}$$

$$d \coloneqq \mathbb{1}_U - \mathbb{1}_K \in \mathbb{R}^V, \tag{20.7}$$

$$\alpha \coloneqq \inf\left\{ \frac{c_e - y\big(\varphi(e)\big)}{d\big(\varphi(e)\big)} : e \in E \setminus E^=(y),\, d\big(\varphi(e)\big) > 0 \right\}. \tag{20.8}$$

If α is finite, then (M, y') also satisfies (20.1), where

$$y' \coloneqq y + \alpha d. \tag{20.9}$$

Proof. By (20.5), the hypotheses of Theorem 8.6 are met for the graph $G^=(y)$; refer back to (20.2) and (20.3). Thus, Theorem 8.6 shows that

$$K \text{ is a vertex cover of } G^=(y) \text{ and } |K| = |M|. \tag{20.10}$$

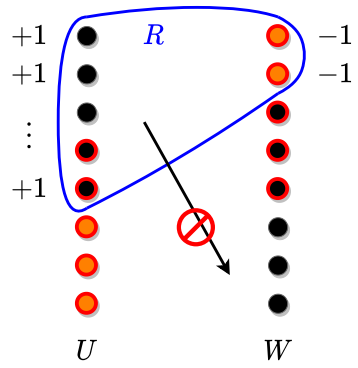


FIGURE 20.1. An illustration for the dual update direction d in relation to the reachable set $R \coloneqq R(M, y)$ in the orientation $D(G^{=}(y), U, W, M)$; the orange vertices lie in the vertex cover K.

Thus, it follows from (19.15) that (20.1)(i) holds for (M, y'). To prove (20.1)(ii), it suffices to show that

$$M \subseteq E^{=}(y'). \tag{20.11}$$

By (8.3) and (20.10)[i], we have

$$|\varphi(e) \cap K| = 1 \qquad \text{for each } e \in M. \tag{20.12}$$

By (19.13) and (20.12),

$$d\big(\varphi(e)\big) = |\varphi(e) \cap U| - 1 = 0 \qquad \text{for each } e \in M. \tag{20.13}$$

Thus, (20.9) shows that $y'\big(\varphi(e)\big) = y\big(\varphi(e)\big) = c_e$ for each $e \in M$. This proves (20.11). □

20.3 Theorem. Restore the context from Proposition 20.2. Then

$$R(M, y) \subsetneq R(M, y'). \tag{20.14}$$

Proof. It may help to refer to Figure 20.1 to follow this proof.

First note that (M, y') satisfies (20.1) by Proposition 20.2. In particular, the expressions $D(M, y')$ and $R(M, y')$ from (20.2) and (20.3) parse. Set

$$R \coloneqq R(M, y).$$

Note that

$$\begin{aligned} d &= \mathbb{1}_U - \mathbb{1}_K && \text{by (20.7)} \\ &= \mathbb{1}_U - \big(\mathbb{1}_{U \setminus R} + \mathbb{1}_{W \cap R}\big) && \text{by (20.6), since } U \cap W = \varnothing \\ &= \mathbb{1}_{U \cap R} - \mathbb{1}_{W \cap R} && \text{since } U \setminus (U \setminus R) = U \cap R. \end{aligned} \tag{20.15}$$

Let us show that,

$$\text{if } e \in E \text{ joins } u \in U \text{ and } w \in W, \text{ then } d\big(\varphi(e)\big) = [u \in U \cap R] - [w \in W \cap R]. \tag{20.16}$$

Let $e \in E$, and let $u \in U$ and $w \in W$ such that

$$\varphi(e) = \{u, w\}. \tag{20.17}$$

Then

$$\begin{aligned} d\big(\varphi(e)\big) &= \mathbb{1}_{\varphi(e)}^{\mathsf{T}} d && \text{by (10.12)} \\ &= \big(e_u + e_w\big)^{\mathsf{T}} d && \text{by (10.9) and (20.17)} \\ &= \big(e_u + e_w\big)^{\mathsf{T}} \big(\mathbb{1}_{U \cap R} - \mathbb{1}_{W \cap R}\big) && \text{by (20.15)} \\ &= e_u^{\mathsf{T}} \mathbb{1}_{U \cap R} - e_u^{\mathsf{T}} \mathbb{1}_{W \cap R} + e_w^{\mathsf{T}} \mathbb{1}_{U \cap R} - e_w^{\mathsf{T}} \mathbb{1}_{W \cap R} && \text{by (10.17b) and (10.17c)} \end{aligned}$$

[i] This equality case did not appear before; my apologies for adding them late, which requires you to reread the proof to verify its correctness.

$$\begin{aligned} &= [u \in U \cap R] - [u \in W \cap R] + [w \in U \cap R] - [w \in W \cap R] && \text{by (10.10)} \\ &= [u \in U \cap R] - [w \in W \cap R] && \text{since } u \notin W \text{ and } w \notin U. \end{aligned}$$

This proves (20.16).

Next we show that

$$R(M, y) \subseteq R(M, y'). \tag{20.18}$$

Let $v \in R(M, y)$. By (20.3), there is $u \in U \setminus V_M$ such that $u \rightsquigarrow_{D(M,y)} v$. Let P be a (u, v)-path in $D(M, y)$. We will prove that

$$a \in E^{=}(y') \qquad \text{for each } a \in A(P). \tag{20.19}$$

This will show that P is a (u, v)-path in $D(M, y')$ by (20.2)[i]. Hence, we shall have $v \in R(M, y')$, thus proving (20.18).

Let

$$e \coloneqq a \in A(P) \subseteq A(D(M, y)) = E^{=}(y). \tag{20.20}$$

Let $u \in U$ and $w \in W$ such that

$$\varphi(e) = \{u, w\}. \tag{20.21}$$

Clearly, $V(P) \subseteq R = R(M, y)$; in particular, $u, w \in R$. Thus,

$$u \in U \cap R \qquad \text{and} \qquad w \in W \cap R.$$

Hence, by (20.16), we have

$$d\big(\varphi(e)\big) = 0. \tag{20.22}$$

Thus,

$$\begin{aligned} y'\big(\varphi(e)\big) &= y\big(\varphi(e)\big) && \text{by (20.9)} \\ &= c_e && \text{by (20.20)}. \end{aligned}$$

Hence, $a = e \in E^{=}(y')$, and this proves (20.19). This concludes the proof of (20.18).

Finally, let $f \in E \setminus E^{=}(y)$ such that

$$d\big(\varphi(f)\big) > 0 \tag{20.23}$$

and

$$\alpha = \frac{c_f - y\big(\varphi(f)\big)}{d\big(\varphi(f)\big)}. \tag{20.24}$$

Such an edge exists by our assumption that α is finite. Since $f \notin E^{=}(y)$, we find from (20.1)(ii) that

$$f \notin M. \tag{20.25}$$

Moreover,

$$\begin{aligned} y'\big(\varphi(f)\big) &= y\big(\varphi(f)\big) + \alpha d\big(\varphi(f)\big) && \text{by (20.9)} \\ &= c_f && \text{by (20.24)}, \end{aligned}$$

whence

$$f \in E^{=}(y'). \tag{20.26}$$

Let $u \in U$ and $w \in W$ such that

$$\varphi(f) = \{u, w\}. \tag{20.27}$$

It follows from (20.23), (20.27), and (20.16) that

$$u \in U \cap R, \tag{20.28}$$

$$w \notin W \cap R. \tag{20.29}$$

[i] Note that the matching M is the same for both orientations $D(M, y)$ and $D(M, y')$.

From (20.28) and (20.18), we find that $u \in R(M, y')$. Thus, by (20.3), there is a path P in $D(M, y')$ from a vertex $u' \in U \setminus V_M$ to u. From (20.26) and (20.25), we know that $\langle u, f, w\rangle$ is a path in $D(M, y')$. Thus, $P \cdot \langle u, f, w\rangle$ is a path in $D(M, y')$, which shows that

$$w \in R(M, y'). \tag{20.30}$$

Hence, from (20.30) and (20.29), we find that $w \in R(M, y') \setminus R(M, y)$. Thus, the inclusion (20.18) is strict. □

20.4 Corollary. Let $G = (V, E, \varphi)$ be a (U, W)-bipartite graph such that $|U| = |W|$, and let $c\colon E \to \mathbb{R}$ be a vector. Set $n := |V|$ and $m := |E|$. Let $\preccurlyeq_{\text{lex}}$ denote the **lexicographic order** on $\mathbb{N} \times \mathbb{N}$ (with respect to $(\mathbb{N}, \leq)$ and $(\mathbb{N}, \leq)$, where $\leq$ is the usual "smaller than or equal to" relation), as in §16.9. Let $\prec_{\text{lex}}$ denote the strict relation from $\preccurlyeq_{\text{lex}}$. Suppose Algorithm 19.2 is run on input (G, U, W, c) and $t \in \mathbb{N}$ is such that both (M_t, y_t) and (M_{t+1}, y_{t+1}) are defined. Then

$$L(M_t, y_t) \prec_{\text{lex}} L(M_{t+1}, y_{t+1}). \tag{20.31}$$

Proof. Immediate from Proposition 20.2 and Theorem 20.3. □

20.5 Exercise. Use Corollary 20.4 to prove that Algorithm 19.2 terminates after at most n^2 iterations. Mimic the proof of Corollary 16.11.

20.6 Back to Min-Cost Circulations. Next, we apply the fundamental paradigm from §11.6 to the Minimum-Cost Circulation Problem 17.13. For future reference, we rewrite the LP for minimum-cost circulation, along with its LP dual. Due to an unfortunate clash of symbols (with cycles), we will adopt the vector k of "kosts", in place of the usual symbol c.

The primal LP is

$$\begin{aligned} \text{Minimize} \quad & k^{\mathsf{T}} f \\ \text{subject to} \quad & f \in \mathbb{R}^A \\ & B_D f = 0, \\ & f \geq \ell, \\ & f \leq u, \end{aligned} \tag{20.32}$$

and its dual is

$$\begin{aligned} \text{Maximize} \quad & \ell^{\mathsf{T}} z_+ + u^{\mathsf{T}} z_- \\ \text{subject to} \quad & B_D^{\mathsf{T}} y + z_+ + z_- = k, \\ & y \in \mathbb{R}^V, \\ & z_+ \in \mathbb{R}^A_+, \\ & z_- \in \mathbb{R}^A_-. \end{aligned} \tag{20.33}$$

By LP Weak Duality (see Theorem 11.2), we have

$$k^{\mathsf{T}} f \geq \ell^{\mathsf{T}} z_+ + u^{\mathsf{T}} z_-, \tag{20.34}$$

whenever $f \in \mathbb{R}^A$ is feasible in (20.32) and $(y, z_+, z_-) \in \mathbb{R}^V \times \mathbb{R}^A_+ \times \mathbb{R}^A_-$ is feasible in (20.33). Also, by the complementary slackness conditions (11.4),

(20.35a) equality holds in (20.34) if and only if

(20.35b) $z_+(a) = 0$ for each $a \in A$ such that $\ell_a < f_a$, and

(20.35c) $z_-(a) = 0$ for each $a \in A$ such that $f_a < u_a$.

20.7 The Host of All Residual Digraphs. We extend the construction from §14.3. We start with a "host" digraph[i] which contains as subdigraphs all potential residual digraphs.

Let $D := (V, A, \varphi)$ be a digraph. Define the digraph[ii]

$$\widehat{D} := \big(V, \widehat{A}, \widehat{\varphi}\big), \tag{20.36}$$

[i] The use of this "host" digraph with come in handy later in the (very technical) analysis of an efficient algorithm for the Minimum-Cost Circulation Problem 17.13.

[ii] The "hat" notation on top of D is meant to remind you that the digraph D was "lifted", or made bigger.

(20.37)

where

$$\widehat{A} \coloneqq A \times \{\pm 1\}, \tag{20.38}$$

and

$$\widehat{\varphi}\big((a,\alpha)\big) \coloneqq \varphi(a)^{\alpha} \qquad \text{for each } (a,\alpha) \in \widehat{A}. \tag{20.39}$$

As before, we call an arbitrary arc (a,α) in $\widehat{A}$ a *forward* arc if $\alpha = +1$ and a *backward* arc if $\alpha = -1$.

Verify that (14.7) can be translated to this context as

$$B_{\widehat{D}} e_{(a,\alpha)} = \alpha B_D e_a \qquad \text{for each arc } (a,\alpha) \in \widehat{A}. \tag{20.40}$$

If $k\colon A \to \mathbb{R}$ is a vector of costs, define $\widehat{k}\colon \widehat{A} \to \mathbb{R}$ as

$$\widehat{k}\big((a,\alpha)\big) \coloneqq \alpha k(a) \qquad \text{for each } (a,\alpha) \in \widehat{A}, \tag{20.41}$$

If $C \coloneqq \langle v_0, (a_1,\alpha_1), v_1, \ldots, (a_p,\alpha_p), v_p\rangle$ is a cycle in $\widehat{D}$, define[i]

$$d_C \coloneqq \sum_{i=1}^{p} \alpha_i e_{a_i} \in \{0,\pm 1\}^A. \tag{20.42}$$

Denote

$$\overleftarrow{B} \coloneqq \big\{\, (a,-\alpha) : (a,\alpha) \in B \big\} \qquad \text{for each } B \subseteq \widehat{A}. \tag{20.43}$$

20.8 Proposition. Restore the context from §20.7. If C is a cycle in $\widehat{D}$, then

$$B_D d_C = 0, \tag{20.44}$$

$$k^{\mathsf{T}} d_C = \widehat{k}(C). \tag{20.45}$$

Proof. Unpack $C =: \langle v_0, (a_1,\alpha_1), v_1, \ldots, (a_p,\alpha_p), v_p\rangle$. Then

$$\begin{aligned}
B_D d_C &= B_D \sum_{i=1}^{p} \alpha_i e_{a_i} && \text{by (20.42)}\\
&= \sum_{i=1}^{p} \alpha_i B_D e_{a_i} && \text{by (10.17c) and induction}\\
&= \sum_{i=1}^{p} B_{\widehat{D}} e_{(a_i,\alpha_i)} && \text{by (20.40)}\\
&= B_{\widehat{D}} \sum_{i=1}^{p} e_{(a_i,\alpha_i)} && \text{by (10.17c) and induction}\\
&= B_{\widehat{D}} \mathbb{1}_C && \text{by (11.12)}\\
&= 0 && \text{by (11.14).}
\end{aligned}$$

This proves (20.44). Also,

$$\begin{aligned}
k^{\mathsf{T}} d_C &= k^{\mathsf{T}} \left(\sum_{i=1}^{p} \alpha_i e_{a_i} \right) && \text{by (20.42)}\\
&= \sum_{i=1}^{p} \alpha_i k^{\mathsf{T}} e_{a_i} && \text{by (10.17c) and induction}\\
&= \sum_{i=1}^{p} \alpha_i k(a_i) && \text{by (10.2) and (10.6)}\\
&= \sum_{i=1}^{p} \widehat{k}(a_i,\alpha_i) && \text{by (20.41)}
\end{aligned}$$

[i] The fact that C is a trail is crucial for the fact that all entries of d_C are either 0 or ± 1. Also, compare this to (14.8).

$$= \widehat{k}(C) \qquad \text{by (2.3).}$$

This proves (20.45). □

20.9 Residual Digraphs for Circulations. Restore the context from §20.7. Let $\ell\colon A \to \mathbb{R}_+$ be a vector and let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function such that $\ell \leq u$. Let $k\colon A \to \mathbb{R}$ be a vector of costs. Let $f \in \mathbb{R}^A$ be a feasible circulation.

Define the subdigraph

$$D(\ell, f, u) \coloneqq (V, A_{\ell,f,u}, \widehat{\varphi}\restriction_{A_{\ell,f,u}}) \tag{20.46}$$

of $\widehat{D}$, where

$$A_{\ell,f,u} \coloneqq \big\{\,(a,+1) : a \in A,\ f_a < u_a\big\} \cup \big\{\,(a,-1) : a \in A,\ f_a > \ell_a\big\} \subseteq \widehat{A}. \tag{20.47}$$

The digraph $D(\ell, f, u)$ is sometimes called the **residual digraph of** f (in D, with respect to ℓ and u).

Also, define $k_{\ell,f,u}\colon A_{\ell,f,u} \to \mathbb{R}$ as

$$k_{\ell,f,u} \coloneqq \widehat{k}\restriction_{A_{\ell,f,u}}. \tag{20.48}$$

The next result provides a line search approach for solving (20.32).

20.10 Theorem. Let $D \coloneqq (V, A, \varphi)$ be a digraph. Let $\ell\colon A \to \mathbb{R}_+$ be a vector and let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function such that $\ell \leq u$. Let $k\colon A \to \mathbb{R}$ be a vector of costs. Let $f \in \mathbb{R}^A$ be a feasible circulation. Unpack $D(\ell, f, u) =: (V, A_{\ell,f,u}, \psi)$ and abbreviate $\tilde{k} \coloneqq k_{\ell,f,u}$, so that $\tilde{k}\colon A_{\ell,f,u} \to \mathbb{R}$. For each arc $(a, \alpha) \in A_{\ell,f,u}$, define

$$u_{\ell,f}\big((a,\alpha)\big) \coloneqq \begin{cases} u_a - f_a & \text{if } \alpha = +1, \\ f_a - \ell_a & \text{if } \alpha = -1. \end{cases} \tag{20.49}$$

From (20.47) we get that

$$u_{\ell,f}(b) > 0 \quad \text{for each arc } b \in A_{\ell,f,u}. \tag{20.50}$$

Moreover,

$$u_{\ell,f} \text{ is integral if } \ell,\ f, \text{ and } u \text{ are integral,} \tag{20.51}$$

where we consider a component $+\infty$ to be an integer. If C is a cycle in the residual digraph $D(\ell, f, u)$ such that $\tilde{k}(C) < 0$, then for each real number λ such that

$$0 \leq \lambda \leq \min_{b \in A(C)} u_{\ell,f}(b), \tag{20.52}$$

the vector $f + \lambda d_C$ is a feasible circulation with cost $k^{\mathsf{T}} f + \lambda \tilde{k}(C)$.

Proof. Let λ as in (20.52), and set

$$f' \coloneqq f + \lambda d_C. \tag{20.53}$$

Note that

$$\begin{aligned} B_D f' &= B_D\big(f + \lambda d_C\big) && \text{by (20.53)} \\ &= B_D f + \lambda B_D d_C && \text{by (10.17c)} \\ &= \lambda B_D d_C && \text{since } f \text{ is a circulation} \\ &= 0 && \text{by (20.44) in Proposition 20.8.} \end{aligned}$$

The remainder of the proof that f' is feasible in (20.32), i.e., that $\ell \leq f' \leq u$ is left as an exercise. The objective value of f' in (20.32) is

$$\begin{aligned} k^{\mathsf{T}} f' &= k^{\mathsf{T}}\big(f + \lambda d_C\big) && \text{by (20.53)} \\ &= k^{\mathsf{T}} f + \lambda k^{\mathsf{T}} d_C && \text{by (10.17c)} \\ &= k^{\mathsf{T}} f + \lambda \widehat{k}(C) && \text{by (20.45) in Proposition 20.8} \\ &= k^{\mathsf{T}} f + \lambda \tilde{k}(C) && \text{by (20.48), since } C \text{ is a cycle in } D(\ell, f, u). \end{aligned}$$

This completes the proof. □

Lecture 21: The Cycle Cancelling Algorithm

21.1 The Nonnegative and Nonpositive Part Operators. For a real number $\mu \in \mathbb{R}$, define

$$[\mu]_+ \coloneqq [\mu]_{+1} \coloneqq \max\{0, \mu\} \geq 0, \tag{21.1a}$$

$$[\mu]_- \coloneqq [\mu]_{-1} \coloneqq \min\{0, \mu\} \leq 0. \tag{21.1b}$$

Verify that

$$\alpha[\mu]_\alpha \geq 0 \qquad \text{for each } \mu \in \mathbb{R} \text{ and } \alpha \in \{\pm 1\}. \tag{21.2}$$

Since

$$\max\{\alpha, \beta\} + \min\{\alpha, \beta\} = \alpha + \beta, \qquad \text{for every } \alpha, \beta \in \mathbb{R}, \tag{21.3}$$

we have

$$\mu = [\mu]_+ + [\mu]_- \qquad \text{for each } \mu \in \mathbb{R}, \tag{21.4a}$$

and it is simple to verify that

$$|\mu| = [\mu]_+ - [\mu]_- \qquad \text{for each } \mu \in \mathbb{R}. \tag{21.4b}$$

We may also apply these operators componentwise on vectors. That is, if $x \in \mathbb{R}^V$ is a vector, then the vectors $[x]_+, [x]_{+1}, [x]_-, [x]_{-1}, |x| \in \mathbb{R}^V$ are defined as

$$\begin{aligned} [x]_+(v) &\coloneqq [x]_{+1}(v) \coloneqq \big[x(v)\big]_+, \\ [x]_-(v) &\coloneqq [x]_{-1}(v) \coloneqq \big[x(v)\big]_-, \qquad \text{for each } v \in V. \\ |x|(v) &\coloneqq |x(v)|, \end{aligned} \tag{21.5}$$

Thus, (21.4) generalize to

$$x = [x]_+ + [x]_-, \tag{21.6a}$$

$$|x| = [x]_+ - [x]_- \tag{21.6b}$$

for each $x \in \mathbb{R}^V$.

21.2 Theorem. Let $D \coloneqq (V, A, \varphi)$ be a digraph. Let $\ell \colon A \to \mathbb{R}_+$ be a vector and let $u \colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function such that $\ell \leq u$. Let $k \colon A \to \mathbb{R}$ be a vector of costs. Let $f \in \mathbb{R}^A$ be a feasible circulation. Unpack $D(\ell, f, u) =: (V, A_{\ell,f,u}, \psi)$ as in (20.46), and abbreviate $\tilde{k} \coloneqq k_{\ell,f,u}$ as in (20.48). Suppose that $y \in \mathbb{R}^V$ is a feasible potential for $D(\ell, f, u)$ with respect to $\tilde{k}$. Define $z_+ \in \mathbb{R}^A_+$ and $z_- \in \mathbb{R}^A_-$ by setting

$$z_+ \coloneqq \big[k - B_D^\mathsf{T} y\big]_+ \in \mathbb{R}^A_+, \tag{21.7a}$$

$$z_- \coloneqq \big[k - B_D^\mathsf{T} y\big]_- \in \mathbb{R}^A_-. \tag{21.7b}$$

Then (y, z_+, z_-) is a feasible solution for (20.33) such that equality holds in (20.34). In particular, f is an optimal solution for (20.32) and (y, z_+, z_-) is an optimal solution for (20.33).

Proof. We have

$$\begin{aligned} z_+ + z_- &= \big[k - B_D^\mathsf{T} y\big]_+ + \big[k - B_D^\mathsf{T} y\big]_- && \text{by (21.7)} \\ &= k - B_D^\mathsf{T} y && \text{by (21.6a)}. \end{aligned}$$

Together with (21.7), this proves that

$$(y, z_+, z_-) \text{ is feasible for (20.33)}. \tag{21.8}$$

The statement that y is a feasible potential for $D(\ell, f, u)$ with respect to $\tilde{k}$ is equivalent to the inequality

$$B^\mathsf{T}_{D(\ell,f,u)} y \leq \tilde{k}; \tag{21.9}$$

see the discussion in §11.10. Let $(a, \alpha) \in A_{\ell,f,u}$. The (a,α)th entry of the LHS of (21.9) is

$$\begin{aligned} (B^\mathsf{T}_{D(\ell,f,u)} y)_{(a,\alpha)} &= e^\mathsf{T}_{(a,\alpha)} B^\mathsf{T}_{D(\ell,f,u)} y && \text{by (10.6)} \\ &= \big(B_{D(\ell,f,u)} e_{(a,\alpha)}\big)^\mathsf{T} y && \text{by (10.20)} \\ &= \alpha \big(B_D e_a\big)^\mathsf{T} y && \text{by (20.40)} \end{aligned}$$

$$= \alpha e_a^\mathsf{T} B_D^\mathsf{T} y \qquad \text{by (10.20)}$$
$$= \alpha\big(B_D^\mathsf{T} y\big)(a) \qquad \text{by (10.6)}.$$

Hence, by (20.41), the inequality (21.9) can be written componentwise as

$$\alpha\Big(\big(k - B_D^\mathsf{T} y\big)(a)\Big) \geq 0 \qquad \text{for each } (a,\alpha) \in A_{\ell,f,u}. \tag{21.10}$$

We shall use (21.10) to verify the sufficient conditions (20.35a) for equality in (20.34). Let $a \in A$ such that $\ell_a < f_a$. Then $(a,-1) \in A_{\ell,f,u}$. Thus, (21.10) applied to $(a,-1)$ shows that

$$\big(k - B_D^\mathsf{T} y\big)(a) \leq 0,$$

whence

$$z_+(a) = \Big[\underbrace{\big(k - B_D^\mathsf{T} y\big)(a)}_{\leq 0}\Big]_+ \qquad \text{by (21.7a) and (21.5)} \tag{21.11}$$
$$= 0 \qquad \text{by (21.1a)}. \tag{21.12}$$

This proves (20.35b).

Next, let $a \in A$ such that $f_a < u_a$. Then $(a,+1) \in A_{\ell,f,u}$. Thus, (21.10) applied to $(a,+1)$ shows that

$$\big(k - B_D^\mathsf{T} y\big)(a) \geq 0,$$

whence

$$z_-(a) = \Big[\underbrace{\big(k - B_D^\mathsf{T} y\big)(a)}_{\geq 0}\Big]_- \qquad \text{by (21.7b) and (21.5)} \tag{21.13}$$
$$= 0 \qquad \text{by (21.1b)}. \tag{21.14}$$

This proves (20.35c). It follows from (20.35a) that equality holds in (20.34). □

21.3 Corollary. Let $D \coloneqq (V, A, \varphi)$ be a digraph. Let $\ell\colon A \to \mathbb{R}_+$ be a vector and let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function such that $\ell \leq u$. Let $k\colon A \to \mathbb{R}$ be a vector of costs. Let $f \in \mathbb{R}^A$ be a feasible circulation. Unpack $D(\ell, f, u) =: (V, A_{\ell,f,u}, \psi)$ as in (20.46), and abbreviate $\tilde{k} \coloneqq k_{\ell,f,u}$ as in (20.48). Then precisely one of the following holds:

(21.15)
- (i) there is a cycle C in the digraph $D(\ell, f, u)$ such that $\tilde{k}(C) < 0$, in which case for each real number $\lambda \geq 0$ bounded above by the positive quantity in the RHS of (20.52), the vector $f + \lambda d_C$ is a feasible circulation with cost $k^\mathsf{T} f + \lambda \tilde{k}(C)$;
- (ii) there is a feasible potential y for the digraph $D(\ell, f, u)$ with respect to the cost vector $\tilde{k}$, in which case $\big(y, \big[k - B_D^\mathsf{T} y\big]_+, \big[k - B_D^\mathsf{T} y\big]_-\big)$ is an optimal solution for (20.33) with the same objective value as that of f in (20.32).

Proof. Immediate from Corollary 5.6 and Theorems 20.10 and 21.2. □

21.4 The Cycle Cancelling Algorithm. Corollary 21.3 suggests Algorithm 21.1 for solving the Minimum-Cost Circulation Problem 17.13. Namely, one starts by finding a feasible circulation using Algorithm 17.2, and then iterate to improve a current feasible circulation along negative cycles in the residual digraph, as in Theorem 20.10. Finding such negative cycles can be done using Algorithm 4.1, as in the proof of Corollary 5.6. When no such cycle exists, Algorithm 4.1 returns a feasible potential in the residual digraph, which is then used to build an optimal solution for (20.33) with the same objective value as the current feasible circulation, by Theorem 21.2; this certifies optimality of the circulation, by LP Weak Duality. For the other outcomes as well, there are accompanying certificates.

21.5 Comparing Cycle Cancelling and Ford-Fulkerson. The Cycle Cancelling Algorithm 21.1 is a generalization of Ford-Fulkerson's Algorithm 14.1 in many ways:

- (i) a form of Algorithm 14.1 is used to obtain an initial feasible circulation;
- (ii) there is the key idea of performing line search on a direction vector found in a residual digraph;

Algorithm 21.1 Cycle Cancelling Algorithm for Minimum-Cost Circulation Problem 17.13

Input: a tuple (D, k, ℓ, u), formed by
(i) a digraph $D = (V, A, \varphi)$,
(ii) a cost vector $k\colon A \to \mathbb{R}$,
(iii) a vector $\ell\colon A \to \mathbb{R}_+$,
(iv) a function $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ such that $\ell \leq u$.

Output: if it terminates, either one of the following, each being an outcome for (MCC(D, k, ℓ, u)) and a certificate:
(i) "infeasible", along with a subset $S \subseteq V$ such that $\ell\big(\delta^{\text{out}}(S)\big) > u\big(\delta^{\text{in}}(S)\big)$, as in Corollary 17.12 (applied to $V \setminus S$);
(ii) "unbounded", along with a vector $f \in \mathbb{R}^A$ and a vector d such that $k^{\mathsf{T}}d < 0$ and $f + \lambda d$ is a feasible circulation for each $\lambda \geq 0$; moreover, f and d are integral provided ℓ and u are integral;
(iii) "optimal", along with a feasible circulation f and a feasible solution (y, z_+, z_-) for (20.33) such that equality holds in (20.34), as in Theorem 21.2; moreover, f is integral provided ℓ and u are integral.

```
1:  function Cycle-Cancelling(D, k, ℓ, u)
2:      Transshipment(·,·,·) ← Transshipment-via-Flows(·,·,·, Edmonds-Karp)
3:      Ret ← Circulation-via-Transshipment(D, ℓ, u, Transshipment)
4:      if Ret is a subset S ⊆ V such that ℓ(δ^out(S)) > u(δ^in(S)) then
5:          return infeasible, and S
6:      f_0 ← Ret                          ▷ f_0 is a feasible circulation, integral if ℓ and u are integral
7:      for t ← 0 to ∞ do
8:          D_t ← D(ℓ, f_t, u), defined as in (20.46)
9:          Define k_t := k_{ℓ,f_t,u} as in (20.48)
10:         if D_t has a negative cycle with respect to k_t then   ▷ Rely on the proof of Corollary 5.6
11:             Let C_t be a cycle in D_t such that k_t(C_t) < 0
12:             d_t := d_{C_t}, defined as in (20.42)              ▷ d_t is integral
13:             u_t ← u_{ℓ,f_t}, defined as in (20.49)
14:             λ_t ← min{ u_t(b) : b ∈ A(C_t) } ∈ (0, +∞]         ▷ and λ_t ∈ ℤ if ℓ and u are integral
15:             if λ_t = +∞ then
16:                 return unbounded, f_t, and d_t
17:             else
18:                 f_{t+1} ← f_t + λ_t d_t                        ▷ f_{t+1} is integral if f_t and λ_t are integral
19:         else
20:             Let y be a feasible potential for D_t with respect to k_t   ▷ As in the proof of Corollary 5.6
21:             Define z_+ and z_− as in (21.7)
22:             return optimal, f_t, and (y, z_+, z_−)
```

(iii) when one reduces a Maximum Flow instance to a Min-Cost Circulation instance as in §18.12 and then runs Algorithm 21.1 (and considering that the call to Edmonds-Karp is bypassed, since $\ell = 0$ shows that 0 is a feasible circulation), then Algorithm 21.1 behaves in essentially the same way as Algorithm 14.1, for any reasonable definition of "way";
(iv) the latter discussion coupled with Exercise 15.7 shows that Algorithm 21.1 *need not terminate*.

21.6 The Goldberg-Tarjan Algorithm. Goldberg and Tarjan proved that, if the cycle from Line 11 in Algorithm 21.1 is chosen to have minimum *mean* cost (which will be defined soon), then one can obtain some form of combinatorial progress to guarantee not only finite termination, but also a polynomial-time algorithm. In the next few lectures, we will study this algorithm, which solves most of the problems we have seen so far.

21.7 Exercise (Reversed Arcs Across Augmentations). Restore the context from Theorem 20.10. Let α denote the RHS of (20.52), and suppose that α is finite. Set $f' := f + \alpha d_C$. Prove that

$$A_{\ell,f',u} \subseteq A_{\ell,f,u} \cup \overleftarrow{A(C)}, \tag{21.16}$$

using the notation from (20.43), and that

$$\text{there is } b \in A(C) \text{ such that } b \notin A_{\ell,f',u}. \tag{21.17}$$

21.8 Theorem. Let $D \coloneqq (V, A, \varphi)$ be a digraph. Let $\ell\colon A \to \mathbb{R}_+$ be a vector and let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function such that $\ell \leq u$. Let $f_0, f_1 \in \mathbb{R}^A$ be feasible circulations. Define $d\colon \widehat{A} \to \mathbb{R}_+$ by

$$d\big((a,\alpha)\big) \coloneqq \alpha\big[(f_1 - f_0)(a)\big]_\alpha \qquad \text{for each } (a,\alpha) \in \widehat{A}. \tag{21.18}$$

Then

$$d \text{ is a circulation in } \widehat{D}, \tag{21.19}$$

$$\operatorname{supp}(d) \subseteq A_{\ell,f_0,u}, \tag{21.20}$$

$$\overleftarrow{\operatorname{supp}(d)} \subseteq A_{\ell,f_1,u}. \tag{21.21}$$

Proof. We have $d \geq 0$ from (21.2), and

$$\begin{aligned}
B_{\widehat{D}} d &= \sum_{b \in \widehat{A}} d_b B_{\widehat{D}} e_b && \text{by (10.27)} \\
&= \sum_{a \in A} \sum_{\alpha \in \{\pm 1\}} d_{(a,\alpha)} B_{\widehat{D}} e_{(a,\alpha)} && \text{by (20.37)} \\
&= \sum_{a \in A} \sum_{\alpha \in \{\pm 1\}} \Big(\alpha\big[(f_1 - f_0)(a)\big]_\alpha\Big)\Big(\alpha B_D e_a\Big) && \text{by (21.18) and (20.40)} \\
&= \sum_{a \in A} B_D e_a \sum_{\alpha \in \{\pm 1\}} \Big(\big[(f_1 - f_0)(a)\big]_\alpha\Big) \\
&= \sum_{a \in A} (f_1 - f_0)_a B_D e_a && \text{by (21.4a)} \\
&= B_D(f_1 - f_0) && \text{by (10.27)} \\
&= B_D f_1 - B_D f_0 && \text{by (10.17c)} \\
&= 0 && \text{since } f_0 \text{ and } f_1 \text{ are circulations.}
\end{aligned}$$

This proves (21.19).

Let $(a,\alpha) \in \operatorname{supp}(d)$. If $\alpha = +1$, then $0 < d_{(a,\alpha)} = f_1(a) - f_0(a)$ by (21.18), so

$$\ell(a) \leq f_0(a) < f_1(a) \leq u(a).$$

Thus, $(a,\alpha) = (a,+1) \in A_{\ell,f_0,u}$ and $(a,-\alpha) = (a,-1) \in A_{\ell,f_1,u}$.

If $\alpha = -1$, then $0 > -d_{(a,\alpha)} = [f_1(a) - f_0(a)]_-$ by (21.18), so

$$\ell(a) \leq f_1(a) < f_0(a) \leq u(a).$$

Thus, $(a,\alpha) = (a,-1) \in A_{\ell,f_0,u}$ and $(a,-\alpha) = (a,+1) \in A_{\ell,f_1,u}$.

This proves (21.20) and (21.21). □

21.9 Problem (The Minimum-Mean Cost Cycle Problem). The **minimum-mean cost cycle problem** is, **given**

(i) a digraph $D = (V, A, \varphi)$,
(ii) a cost function $k\colon A \to \mathbb{R}$,

solve the optimization problem

$$\text{(M}\mu\text{C}(D,k)) \qquad \begin{aligned} \text{Minimize} \quad & \frac{k(C)}{|C|} \\ \text{subject to} \quad & C \text{ is a cycle in } D; \end{aligned}$$

we denote

$$|C| \coloneqq |V(C)| = |A(C)| \qquad \text{for each cycle } C \tag{21.22}$$

throughout our discussion of this problem.

By Theorem 1.5,

(21.23) either (MμC(D,k)) is infeasible or it has an optimal solution.

Let $\mu(D,k)$ denote the optimal value of (MμC(D,k)), and note that[i]

(21.24) $\mu(D,k) \geq 0$ if and only if D has no negative cycles (with respect to k).

Lecture 22: Karp's Algorithm for Minimum-Mean Cost Cycle

22.1 The Shortest Walk Problem with Prescribed Length Ending at a Vertex. Let $D \coloneqq (V,A,\varphi)$ be a digraph, let $k\colon A \to \mathbb{R}$ be a cost function. Define, for each $v \in V$ and[ii] $\ell \in \mathbb{N}$, the optimization problem

(SWP$_{=\ell}(D,k,v)$)
$$\begin{array}{ll} \text{Minimize} & k(W) \\ \text{subject to} & W \text{ is a walk in } D \text{ ending at } v \text{ of length } \ell. \end{array}$$

By Theorem 1.5,

(22.1) either (SWP$_{=\ell}(D,k,v)$) is infeasible or it has an optimal solution.

Let $d_{k,\ell}(v)$ denote the optimal value of (SWP$_{=\ell}(D,k,v)$). Such optimal values (along with corresponding optimal solutions, when they exist) can be computed by a straightforward dynamic programming recurrence as follows:

(22.2)
$$\left.\begin{aligned} d_{k,0}(v) &= 0 \\ d_{k,\ell+1}(v) &= \inf\{\, d_{k,\ell}(u) + k(a) : a \in \delta^{\mathrm{in}}(v),\ uv \coloneqq \varphi(a)\}\end{aligned}\right\} \quad \text{for each } v \in V,\ \ell \in \mathbb{N}, \text{ and } k \in \mathbb{R}^A.$$

We will (eventually) prove that

(22.3)
$$\mu(D,k) = \min_{v\in V} \sup\left\{ \frac{d_{k,n}(v) - d_{k,\ell}(v)}{n-\ell} : 0 \leq \ell < n \text{ integer},\ d_{k,\ell}(v) < +\infty \right\},$$

where $n \coloneqq |V|$. We will also see how to extract an optimal solution for (MμC(D,k)) from optimal solutions for the RHS of (22.3) and (SWP$_{=\ell}(D,k,v)$); the latter can be readily obtained using dynamic programming.

22.2 Objective Function Shifts. Let $D \coloneqq (V,A,\varphi)$ be a digraph, let $k\colon A \to \mathbb{R}$ be a cost function. Let $\alpha \in \mathbb{R}$. Then for each $\ell \in \mathbb{N}$ and $v \in V$,

(22.4a) $$\mu(D, k+\alpha\mathbb{1}) = \mu(D,k) + \alpha,$$

(22.4b) $$d_{k+\alpha\mathbb{1},\ell}(v) = d_{k,\ell}(v) + \alpha\ell.$$

Moreover,

(22.4c) the optimal solutions for (MμC$(D,k+\alpha\mathbb{1})$) and (MμC(D,k)) are the same, and

(22.4d) the optimal solutions for (SWP$_{=\ell}(D,k+\alpha\mathbb{1},v)$) and (SWP$_{=\ell}(D,k,v)$) are the same.

Proof. Let us prove (22.4a) and (22.4c). If D has no cycles, both sides of (22.4a) are $+\infty$ and both sets of optimal solutions in (22.4c) are empty. So suppose that D has some cycle, and let C be an optimal solution for (MμC(D,k)), which exists by (21.23). The objective value of C in (MμC$(D,k+\alpha\mathbb{1})$) is[iii]

$$\frac{(k+\alpha\mathbb{1})(C)}{|C)|} = \frac{k(C)+\alpha\mathbb{1}(C)}{|C|} = \frac{k(C)+\alpha|C|}{|C|} = \frac{k(C)}{|C|} + \alpha = \mu(D,k) + \alpha,$$

whence

(22.5) $$\mu(D, k+\alpha\mathbb{1}) \leq \mu(D,k) + \alpha.$$

Set $k' \coloneqq k + \alpha\mathbb{1}$ and $\alpha' \coloneqq -\alpha$, and apply (22.5) again to obtain

$$\mu(D, k' + \alpha'\mathbb{1}) \leq \mu(D,k') + \alpha',$$

i.e.,

(22.6) $$\mu(D,k) \leq \mu(D, k+\alpha\mathbb{1}) - \alpha.$$

---

[i] An important qualitative difference with (SWP) is that, there, or more precisely in Corollary 5.6, the optimal value is *infinite* if and only if there is a negative cycle. Here, existence of a negative cycle is detected by a *sign* in a *finite* quantity.

[ii] We are temporarily moving away from feasible circulations throughout our discussion of Problem 21.9, so we will make use of the symbol ℓ for lengths of walks, and u may refer to a vertex.

[iii] Recall our abuse of notation from (21.22)

Then (22.5) and (22.6) prove (22.4a) and (22.4c).

The proof of (22.4b) and (22.4d) is similar, as an easy induction on ℓ using (22.2). □

22.3 Theorem. Let $D \coloneqq (V, A, \varphi)$ be a digraph on n vertices, let $k\colon A \to \mathbb{R}$ be a cost function. Suppose that

$$\min_{u\in V} \sup\left\{ \frac{d_{k,n}(u) - d_{k,\ell}(u)}{n-\ell} : 0 \le \ell < n \text{ integer}, d_{k,\ell}(u) < +\infty \right\} \tag{22.7}$$

is finite, and attained by $v \in V$. Let W be an optimal solution for (SWP$_{=n}(D, k, v)$); existence of W is guaranteed by finiteness of (22.7). Unpack $W =: \langle v_0, \dots, a_n, v_n\rangle$. If $i, j \in \{0, \dots, n\}$ are such that $i < j$ and $C \coloneqq \langle v_i, \dots, a_j, v_j\rangle$ is a cycle, then

$$\frac{k(C)}{|C|} \le \min_{u\in V} \sup\left\{ \frac{d_{k,n}(u) - d_{k,\ell}(u)}{n-\ell} : 0 \le \ell < n \text{ integer}, d_{k,\ell}(u) < +\infty \right\}. \tag{22.8}$$

Proof. Suppose $W_{ij} \coloneqq C = \langle v_i, \dots, a_j, v_j\rangle$ is a cycle. Thus, $v_i = v_j$. Set $\ell \coloneqq n - (j - i)$, so that

$$0 \le \ell < n, \tag{22.9}$$

$$W_{ij} = C \text{ has length } j - i = n - \ell, \tag{22.10}$$

$$W'_{ij} \coloneqq \langle v_0, \dots, a_i, v_i\rangle \cdot \langle v_j, \dots, a_n, v_n\rangle \text{ has length } n - (j - i) = \ell. \tag{22.11}$$

From (22.11) we obtain

$$d_{k,\ell}(v) \le k(W'_{ij}). \tag{22.12}$$

We have

$$\begin{aligned} d_{k,n}(v) &= k(W) && \text{since } W \text{ is optimal for (SWP}_{=n}(D, k, v)) \\ &= k(W_{ij}) + k(W'_{ij}) \\ &\ge k(C) + d_{k,\ell}(v) && \text{by (22.12)}. \end{aligned} \tag{22.13}$$

Thus,

$$\begin{aligned} \frac{k(C)}{|C|} &= \frac{k(C)}{n-\ell} && \text{by (22.10)} \\ &\le \frac{d_{k,n}(v) - d_{k,\ell}(v)}{n-\ell} && \text{by (22.13)} \\ &\le \sup\left\{ \frac{d_{k,n}(v) - d_{k,\ell}(v)}{n-\ell} : 0 \le \ell < n \text{ integer}, d_{k,\ell}(v) < +\infty \right\} && \text{by (22.9) and (22.12)}, \end{aligned}$$

the latter of which is equal to the RHS of (22.8) by the definition of v. □

22.4 Lemma. Let $D \coloneqq (V, A, \varphi)$ be a digraph, let $k\colon A \to \mathbb{R}$ be a cost function. Let $s \in V$. Then the optimal value of

$$\begin{aligned} &\text{Minimize} && k(W) \\ &\text{subject to} && W \text{ is a walk in } D \text{ ending at } s. \end{aligned} \tag{22.14}$$

is equal to $\inf_{\ell\in\mathbb{N}} d_{k,\ell}(s)$ and, if D has no negative cycles, then there is a path P in D ending at s which is an optimal solution for (22.14).

Proof. The first statement of the result is a simple exercise. Suppose that

$$D \text{ has no negative cycles.} \tag{22.15}$$

We will show that (22.14) is (homomorphically) equivalent to

$$\begin{aligned} &\text{Minimize} && k(P) \\ &\text{subject to} && P \text{ is a path in } D \text{ ending at } s. \end{aligned} \tag{22.16}$$

By Theorem 1.5, the optimization problem (22.16) has an optimal solution. This shall complete the proof of the second statement of the result, via (6.6)(iv). Clearly, the identity map is a homomorphism from (22.16) to (22.14).

Let W be a feasible solution for (22.14), and suppose W starts at $r \in V$. Apply Theorem 3.1 to obtain a path P from r to s such that $k(P) \le k(W)$; note that (3.1)(ii) is ruled out by the hypothesis (22.15). Thus,

we may set $\psi(W) \coloneqq P$. By doing this for every feasible solution W for (22.14), the function ψ thus obtained is a homomorphism from (22.14) to (22.16). □

22.5 Corollary. Let $D \coloneqq (V, A, \varphi)$ be a digraph on n vertices, let $k\colon A \to \mathbb{R}$ be a cost function. Let $s \in V$. If $\mu(D, k) = 0$, and C is a cycle in D of length ℓ with $k(C) = 0$ and $s \in V(C)$, then there is an optimal solution W for (22.14) which is a walk whose length h satisfies $n - \ell \leq h < n$.

Proof. From (21.24), there are no negative cycles in D. Let W_0 be an optimal solution for (22.14) which is a path, which exists by Lemma 22.4. Let ℓ_0 be the length of W_0; note that

$$0 \leq \ell_0 < n. \tag{22.17}$$

We may assume that $C = \langle v_0, \ldots, a_\ell, v_\ell \rangle$ with $v_0 = v_\ell = s$. Define inductively $W_{p+1} \coloneqq W_p \cdot C$ for each $p \in \mathbb{N}$. Note that each W_p is an optimal solution for (22.14) with length $\ell_0 + p\ell$. Set

$$\bar{p} \coloneqq \left\lceil \frac{n - \ell_0}{\ell} \right\rceil - 1; \tag{22.18}$$

note that $\bar{p} \geq 0$ by (22.17). From

$$\alpha \leq \lceil \alpha \rceil < \alpha + 1 \qquad \text{for each } \alpha \in \mathbb{R},$$

applied to $\alpha \coloneqq (n - \ell_0)/\ell$, we obtain that the length $h \coloneqq \ell_0 + \bar{p}\ell$ of $W_{\bar{p}}$ satisfies

$$n - \ell = \ell_0 + \ell\Big(\frac{n - \ell_0}{\ell} - 1\Big) \leq \underbrace{\ell_0 + \ell\bar{p}}_{\| \; h} < \ell_0 + \ell\Big(\frac{n - \ell_0}{\ell}\Big) = n. \qquad \square$$

22.6 Theorem. Let $D \coloneqq (V, A, \varphi)$ be a digraph on n vertices, let $k\colon A \to \mathbb{R}$ be a cost function. Then

$$\mu(D, k) \geq \min_{u \in V} \sup \left\{ \frac{d_{k,n}(u) - d_{k,\ell}(u)}{n - \ell} : 0 \leq \ell < n \text{ integer}, \, d_{k,\ell}(u) < +\infty \right\}. \tag{22.19}$$

Proof. If $\mu(D, k) = +\infty$, there is nothing to prove, so we may assume that

$$\alpha \coloneqq \mu(D, k) < +\infty. \tag{22.20}$$

Set

$$k' \coloneqq k - \alpha \mathbb{1}. \tag{22.21}$$

Then

$$\begin{aligned} \mu(D, k') &= \mu(D, k - \alpha\mathbb{1}) && \text{by (22.21)} \\ &= \mu(D, k) - \alpha && \text{by (22.4a)} \\ &= 0 && \text{by (22.20).} \end{aligned} \tag{22.22}$$

Thus, by (22.22) and (21.24),

$$D \text{ has no negative cycles with respect to } k'. \tag{22.23}$$

By (21.23) and (22.20), there is a cycle $C \coloneqq \langle v_0, \ldots, a_t, v_t \rangle$ in D such that $k'(C) = \mu(D, k')$, so that

$$k'(C) = 0 \tag{22.24}$$

by (22.22). Set

$$s \coloneqq v_0 = v_t.$$

By (22.22) and Corollary 22.5, there is an optimal solution W for (22.14) (with k') which is a walk whose length h satisfies

$$0 \leq n - t \leq h < n. \tag{22.25}$$

Thus, by Lemma 22.4,

$$k'(W) = \min_{\ell \in \mathbb{N}} d_{k',\ell}(s). \tag{22.26}$$

Note that

$$W \text{ is a walk ending at } s \text{ of length } h \text{ and } k'(W) = d_{k',h}(s). \tag{22.27}$$

From (22.25), we have $0 < n - h \leq t \leq n$. Set

$$v \coloneqq v_{n-h}, \tag{22.28}$$

$$P \coloneqq \langle \underbrace{v_0}_{\| s}, \ldots, a_{n-h}, \underbrace{v_{n-h}}_{\| v} \rangle, \text{ an } (s,v)\text{-walk of length } n-h, \text{ and} \tag{22.29}$$

$$Q \coloneqq \langle \underbrace{v_{n-h}}_{\| v}, \ldots, a_t, \underbrace{v_t}_{\| s} \rangle, \text{ a } (v,s)\text{-walk of length } t-(n-h), \tag{22.30}$$

so that

$$C = P \cdot Q. \tag{22.31}$$

By (22.27) and (22.29), the walk $W \cdot P$ is a feasible solution for (SWP$_{=n}(D, k', v)$), so

$$\begin{aligned} d_{k',n}(v) &\leq k'(W \cdot P) \\ &= k'(W) + k'(P). \end{aligned} \tag{22.32}$$

From (22.32), the optimal value $d_{k',n}(v)$ is finite. Hence,

$$d_{k',\ell}(v) < +\infty \qquad \text{for each } \ell \in \mathbb{N} \text{ such that } 0 \leq \ell \leq n. \tag{22.33}$$

We will prove that

$$0 \geq d_{k',n}(v) - d_{k',\ell}(v) \qquad \text{for each } \ell \in \mathbb{N} \text{ such that } 0 \leq \ell < n. \tag{22.34}$$

Let $\ell \in \mathbb{N}$ such that $0 \leq \ell < n$. By (22.33), the optimization problem (SWP$_{=\ell}(D, k', v)$) has an optimal solution. Let

$$W' \text{ be an optimal for (SWP}_{=\ell}(D, k', v)). \tag{22.35}$$

Then

$$W' \cdot Q \text{ is feasible for (SWP}_{=\ell+t-(n-h)}(D, k', s)), \tag{22.36}$$

whence

$$\begin{aligned} d_{k',\ell}(v) + k'(Q) &= k'(W') + k'(Q) && \text{by (22.35)} \\ &= k'(W' \cdot Q) \\ &\geq d_{k',\ell+t-(n-h)}(s) && \text{by (22.36)} \\ &\geq k'(W) && \text{by (22.26)} \\ &\geq d_{k',n}(v) - k'(P) && \text{by (22.32)}. \end{aligned} \tag{22.37}$$

Thus,

$$\begin{aligned} 0 &= k'(C) && \text{by (22.24)} \\ &= k'(P) + k'(Q) && \text{by (22.31)} \\ &\geq d_{k',n}(v) - d_{k',\ell}(v) && \text{by (22.37)}. \end{aligned}$$

This completes the proof of (22.34).

It follows easily from (22.34) that

$$\mu(D,k) \geq \frac{d_{k,n}(v) - d_{k,\ell}(v)}{n-\ell} \qquad \text{for each } \ell \in \mathbb{N} \text{ such that } 0 \leq \ell < n. \tag{22.38}$$

Indeed, if $\ell \in \mathbb{N}$ is such that $0 \leq \ell < n$, then

$$\begin{aligned} 0 &\geq \frac{d_{k',n}(v) - d_{k',\ell}(v)}{n-\ell} && \text{by (22.34)} \\ &= \frac{d_{k,n}(v) - \alpha n - d_{k,\ell}(v) + \alpha \ell}{n-\ell} && \text{by (22.21) and (22.4b)} \\ &= \frac{d_{k,n}(v) - d_{k,\ell}(v)}{n-\ell} - \alpha \\ &= \frac{d_{k,n}(v) - d_{k,\ell}(v)}{n-\ell} - \mu(D,k) && \text{by (22.20)}. \end{aligned}$$

By (22.4b), the optimal value $d_{k',\ell}(v)$ is finite if and only if $d_{k,\ell}(v)$ is finite. Thus, from (22.38) we get

$$\begin{aligned}\mu(D,k) &\geq \sup\left\{ \frac{d_{k,n}(v) - d_{k,\ell}(v)}{n-\ell} : 0 \leq \ell < n \text{ integer}, d_{k,\ell}(v) < +\infty \right\} \\ &\geq \min_{u\in V} \sup\left\{ \frac{d_{k,n}(u) - d_{k,\ell}(u)}{n-\ell} : 0 \leq \ell < n \text{ integer}, d_{k,\ell}(u) < +\infty \right\}. \qquad \square\end{aligned}$$

22.7 Corollary. Let $D \coloneqq (V, A, \varphi)$ be a digraph on n vertices, let $k\colon A \to \mathbb{R}$ be a cost function. Then

$$\mu(D,k) = \min_{v\in V} \sup\left\{ \frac{d_{k,n}(v) - d_{k,\ell}(v)}{n-\ell} : 0 \leq \ell < n \text{ integer}, d_{k,\ell}(v) < +\infty \right\}. \tag{22.39}$$

Proof. Immediate from Theorems 22.3 and 22.6. $\square$

22.8 Karp's Minimum-Mean Cost Cycle Algorithm. Algorithm 22.1 solves (MμC(D,k)) using the dynamic programming recurrence (22.2), Corollary 22.7 and Theorem 22.3. Namely, optimal values/solutions for (SWP$_{=\ell}(D,k,v)$) are computed for every length up to n and each ending vertex v. Then the optimal value $\mu(D,k)$ for (MμC(D,k)) is computed using the formula from Corollary 22.7. Finally, if the optimal value is finite, an optimal solution is extracted by using Theorem 22.3.

Algorithm 22.1 Karp's Minimum-Mean Cycle Algorithm

Input: a tuple (D,k), formed by
(i) a digraph $D =: (V, A, \varphi)$,
(ii) a cost vector $k\colon A \to \mathbb{R}$.
Output: the outcome and an optimal solution for (MμC(D,k)) if one exists.

function KARP-MIN-MEAN-CYCLE(D,k)
  for all $v \in V$ **do**
    $d_0(v) \leftarrow 0$
    $W_0(v) \leftarrow \langle v \rangle$
  for $\ell \leftarrow 0$ **to** $n-1$ **do**
    for all $v \in V$ **do**
      if there is $a \in \delta^{\text{in}}(v)$ with tail u such that $d_\ell(u) < +\infty$ **then**
        Let a attain $\min\{ d_\ell(u) + k(a) : a \in \delta^{\text{in}}(v),\ uv \coloneqq \varphi(a) \}$
        $uv \leftarrow \varphi(a)$
        $d_{\ell+1}(v) \leftarrow d_\ell(u) + k(a)$
        $W_{\ell+1}(v) \leftarrow W_\ell(u) \cdot \langle u, a, v \rangle$
      else
        $d_{\ell+1}(v) \leftarrow \infty$
        $W_{\ell+1}(v) \leftarrow \perp$
  Let v attain $\min_{u\in V} \sup\left\{ \frac{d_n(u) - d_\ell(u)}{n-\ell} : 0 \leq \ell < n \text{ integer}, d_\ell(u) < +\infty \right\}$
  if $\sup\left\{ \frac{d_n(v) - d_\ell(v)}{n-\ell} : 0 \leq \ell < n \text{ integer}, d_\ell(v) < +\infty \right\} = +\infty$ **then**
    return infeasible
  else
    Unpack $\langle v_0, \ldots, a_n, v_n \rangle \leftarrow W_n(v)$
    Find $i, j \in \{0, \ldots, n\}$ such that $i < j$ and $\langle v_i, \ldots, a_j, v_j \rangle$ is a cycle
    return $\langle v_i, \ldots, a_j, v_j \rangle$

LECTURE 23: THE MINIMUM-MEAN COST CYCLE CANCELLING ALGORITHM

23.1 Exercise (Minimum-Mean Cost Cycles, Feasible Potentials, and LPs). Let $D \coloneqq (V, A, \varphi)$ be a digraph and let $k\colon A \to \mathbb{R}$ be a cost function. Let $\alpha \in \mathbb{R}$. We have

$$\begin{aligned}\mu(D,k) \geq -\alpha &\iff \mu(D,k) + \alpha \geq 0 \\ &\iff \mu(D, k + \alpha\mathbb{1}) \geq 0 \qquad \text{by (22.4a)}\end{aligned}$$

$$\begin{aligned}
&\iff D \text{ has no negative cycles w.r.t. } k+\alpha\mathbb{1} && \text{by (21.24)}\\
&\iff \text{there is } y\in\mathbb{R}^V \text{ such that } k+\alpha\mathbb{1}\geq B_D^{\mathsf{T}}y; && \text{see §11.10.}
\end{aligned}$$

Conclude that $-\mu(D,k)$ is the optimal value of the LP

$$\begin{aligned}
\text{Minimize}\quad & \alpha\\
\text{subject to}\quad & k+\alpha\mathbb{1}\geq B_D^{\mathsf{T}}y,\\
& y\in\mathbb{R}^V,\\
& \alpha\in\mathbb{R}.
\end{aligned}\tag{23.1}$$

23.2 Proposition. Restore the context from Exercise 23.1. If C is an optimal solution for (MμC(D,k)) and (y,α) is an optimal solution for (23.1), then

$$\big(k+\alpha\mathbb{1}\big)(a)=\big(B_D^{\mathsf{T}}y\big)(a)\qquad\text{for each } a\in A(C).\tag{23.2}$$

Proof. By Exercise 23.1, we have

$$\alpha=-\mu(D,k)=-\frac{k(C)}{|C|}.\tag{23.3}$$

Since $k+\alpha\mathbb{1}\geq B_D^{\mathsf{T}}y$ and $\mathbb{1}_C\geq 0$, we find from (11.1)(iv) that

$$\mathbb{1}_C^{\mathsf{T}}(k+\alpha\mathbb{1})\geq\mathbb{1}_C^{\mathsf{T}}B_D^{\mathsf{T}}y,\tag{23.4}$$

and

$$\text{equality holds in (23.4) if and only if (23.2) holds.}\tag{23.5}$$

The LHS of (23.4) is

$$\begin{aligned}
\mathbb{1}_C^{\mathsf{T}}(k+\alpha\mathbb{1}) &= \mathbb{1}_C^{\mathsf{T}}k+\alpha\mathbb{1}_C^{\mathsf{T}}\mathbb{1} && \text{by (10.17c)}\\
&= k(C)+\alpha|C| && \text{by (11.13)}\\
&= 0 && \text{by (23.3).}
\end{aligned}\tag{23.6}$$

The RHS of (23.4) is

$$\begin{aligned}
\mathbb{1}_C^{\mathsf{T}}B_D^{\mathsf{T}}y &= y^{\mathsf{T}}B_D\mathbb{1}_C && \text{by (10.20)}\\
&= 0 && \text{by (11.14).}
\end{aligned}\tag{23.7}$$

From (23.6) and (23.7), equality holds in (23.4), so (23.5) completes the proof. □

23.3 The Minimum-Mean Cost Cycle Cancelling Algorithm. The instantiation of the Cycle Cancelling Algorithm 21.1 using Karp's Minimum-Mean Cost Cycle Algorithm 22.1 appears as Algorithm 23.1. It is straightforward to verify that each iteration can be made to run in polynomial time. We proceed to bound the number of iterations as a polynomial of the input size.

The analysis is fairly technical and long[1], and it differs from previous algorithmic analyses in many respects:

(i) We do *not* focus on the progress obtained only from an iteration to the next; rather, we will analyze progress made across *multiple iterations*.

(ii) It will be very useful to skew/perturb the (iterate) cost function k_t from Line 9 in Algorithm 23.1 by some feasible potential y for the residual digraph D_t; namely, it will be helpful to look at the perturbed cost function $k_t - B_{D_t}^{\mathsf{T}}y$.

(iii) Not only will we perturb the cost function k_t from iteration t, but in fact we will need to look at (perturbed) costs of arcs *not* in the current residual digraph D_t. For that, we will use the host of all residual digraphs from §20.7 and its corresponding cost function.

[1]The arguments are not used elsewhere in the course, and we show the whole proof as it is a culmination of many algorithms and techniques presented for previous problems. Your understanding (or lack thereof) of this proof will *not* be tested.

Algorithm 23.1 Minimum-Mean Cost Cycle Cancelling Algorithm

Input: same as in Algorithm 21.1
Output: same as in Algorithm 21.1

function MIN-MEAN-COST-CYCLE-CANCELLING(D, k, ℓ, u)
  TRANSSHIPMENT$(\cdot,\cdot,\cdot) \leftarrow$ TRANSSHIPMENT-VIA-FLOWS$(\cdot,\cdot,\cdot,$ EDMONDS-KARP$)$
  Ret $\leftarrow$ CIRCULATION-VIA-TRANSSHIPMENT$(D, \ell, u,$ TRANSSHIPMENT$)$
  if Ret is a subset $S \subseteq V$ such that $\ell\big(\delta^{\text{out}}(S)\big) > u\big(\delta^{\text{in}}(S)\big)$ **then**
    return infeasible, and S
  $f_0 \leftarrow$ Ret
  for $t \leftarrow 0$ **to** ∞ **do**
    $D_t \leftarrow D(\ell, f_t, u)$, defined as in (20.46)
    Define $k_t \coloneqq k_{\ell, f_t, u}$ as in (20.48)
    $\text{Ret}_t \leftarrow$ KARP-MIN-MEAN-CYCLE(D_t, k_t)
    if Ret_t is infeasible **then**
      Let y be a feasible potential for D_t with respect to k_t
      return optimal, f_t, and (y, z_+, z_-), with z_+ and z_- defined as in (21.7)
    $C_t \leftarrow \text{Ret}_t$
    if $k_t(C_t) \geq 0$ **then**
      Let y be a feasible potential for D_t with respect to k_t
      return optimal, f_t, and (y, z_+, z_-), with z_+ and z_- defined as in (21.7)
    $d_t \coloneqq d_{C_t}$, defined as in (20.42)
    $u_t \leftarrow u_{\ell, f_t}$, defined as in (20.50)
    $\lambda_t \leftarrow \min\big\{ u_t(b) : b \in A(C_t)\big\} \in (0, +\infty]$
    if $\lambda_t = +\infty$ **then**
      return unbounded, f_t, and d_t
    else
      $f_{t+1} \leftarrow f_t + \lambda_t d_t$

23.4 The Running Context. Let $D \coloneqq (V, A, \varphi)$ be a digraph on $n \geq 2$ vertices and m arcs. Let $\ell\colon A \to \mathbb{R}_+$ be a vector and let $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$ be a function such that $\ell \leq u$. Let $k\colon A \to \mathbb{R}$ be a vector of costs. Suppose that Algorithm 23.1 is run on input (D, k, ℓ, u), and let $T \subseteq \mathbb{N}$ denote the set of indices t for which f_t is defined. Recall the notation set in §20.7.

Let $t \in T$. Define[1]

$$\widehat{B} \coloneqq B_{\widehat{D}}, \tag{23.8}$$

$$\varepsilon_t \coloneqq -\mu(D_t, k_t), \tag{23.9}$$

$$A_t \coloneqq A(D_t), \tag{23.10}$$

$$B_t \coloneqq B_{D_t}, \tag{23.11}$$

Define the LP

$$(\text{M}\mu\text{C}_t) \qquad \begin{aligned} \text{Minimize} \quad & \alpha \\ \text{subject to} \quad & k_t - B_t^{\mathsf{T}} y \geq -\alpha \mathbb{1}, \\ & y \in \mathbb{R}^V, \\ & \alpha \in \mathbb{R}; \end{aligned}$$

Note that $(\text{M}\mu\text{C}_t)$ is the same LP as (23.1) applied to (D_t, k_t). Thus, by Exercise 23.1 and (23.9),

$$\varepsilon_t \text{ is the optimal value of } (\text{M}\mu\text{C}_t). \tag{23.12}$$

[1] The quantity ε_t can be seen as the progress obtained at each iteration. We will see that the sequence of ε_t's is a sequence of positive (thus the sign change in (23.9)) numbers that decrease to zero. When it reaches zero, the algorithm terminates, since then no negative cycle exists by (21.24).

From Line 9 in Algorithm 23.1, we have

$$k_t \colon A_t \to \mathbb{R}, \tag{23.13}$$

with

$$k_t\big((a,\alpha)\big) = \widehat{k}\big((a,\alpha)\big) = \alpha k(a) \qquad \text{for each } (a,\alpha) \in A_t. \tag{23.14}$$

(23.15)

If T is bounded, set $T' \coloneqq T \setminus \{\max T\}$, i.e., T' has all *but* the index of the last iteration. Otherwise, just set $T' \coloneqq T$. Let $t \in T'$. Then

$$C_t \text{ is an optimal solution for } (\mathrm{M}\mu\mathrm{C}(D_t, k_t)), \tag{23.16}$$

$$\varepsilon_t > 0. \tag{23.17}$$

By Exercise 23.1, there is $y_t \in \mathbb{R}^V$ such that

$$(y_t, \varepsilon_t) \text{ is an optimal solution for } (\mathrm{M}\mu\mathrm{C}_t). \tag{23.18}$$

Define

$$\widehat{k}_t \coloneqq \widehat{k} - \widehat{B}^{\mathsf{T}} y_t \in \mathbb{R}^{\widehat{A}}. \tag{23.19}$$

By Exercise 21.7,

$$A_{t+1} \subseteq A_t \cup \overleftarrow{A(C_t)}, \tag{23.20}$$

and

$$\text{there is } b \in A(C_t) \text{ such that } b \notin A_{t+1}. \tag{23.21}$$

23.5 Lemma. Restore the context from §23.4. Let $t \in T$ and let $y \in \mathbb{R}^V$. Then:

$$\big(\widehat{k} - \widehat{B}^{\mathsf{T}} y\big)\restriction_{A_t} = k_t - B_t^{\mathsf{T}} y, \tag{23.22a}$$

$$\big(\widehat{k} - \widehat{B}^{\mathsf{T}} y\big)\big((a,-\alpha)\big) = -\big(\widehat{k} - \widehat{B}^{\mathsf{T}} y\big)\big((a,\alpha)\big) \qquad \text{for each } (a,\alpha) \in \widehat{A}, \tag{23.22b}$$

$$\big(\widehat{k} - \widehat{B}^{\mathsf{T}} y\big)(C) = \widehat{k}(C) \qquad \text{for each cycle } C \text{ of } \widehat{D}. \tag{23.22c}$$

In particular, if $t \in T'$, then:

$$\widehat{k}_t\restriction_{A_t} = k_t - B_t^{\mathsf{T}} y_t, \tag{23.23a}$$

$$\widehat{k}_t\big((a,-\alpha)\big) = -\widehat{k}_t\big((a,\alpha)\big) \qquad \text{for each } (a,\alpha) \in \widehat{A}, \tag{23.23b}$$

$$\widehat{k}_t(C) = \widehat{k}(C) \qquad \text{for each cycle } C \text{ of } \widehat{D}. \tag{23.23c}$$

Furthermore,

$$\widehat{k}_t\big((a,\alpha)\big) \geq -\varepsilon_t \qquad \text{for each } (a,\alpha) \in A_t, \text{ with equality if } (a,\alpha) \in A(C_t). \tag{23.24}$$

Proof. Let $(a,\alpha) \in A_t$. Then

$$\begin{aligned}
(\widehat{B}^{\mathsf{T}} y)_{(a,\alpha)} &= e_{(a,\alpha)}^{\mathsf{T}} B_{\widehat{D}}^{\mathsf{T}} y && \text{by (10.6) and (23.8)} \\
&= y^{\mathsf{T}} B_{\widehat{D}} e_{(a,\alpha)} && \text{by (10.20)} \\
&= y^{\mathsf{T}} B_t e_{(a,\alpha)} && \text{by (20.46) and (20.47)} \\
&= e_{(a,\alpha)}^{\mathsf{T}} B_t^{\mathsf{T}} y && \text{by (10.20)} \\
&= (B_t^{\mathsf{T}} y)_{(a,\alpha)} && \text{by (10.6).}
\end{aligned}$$

Together with (23.14), this proves (23.22a).

Let $(a,\alpha) \in \widehat{A}$. Then

$$\begin{aligned}
(\widehat{B}^{\mathsf{T}} y)_{(a,-\alpha)} &= e_{(a,-\alpha)}^{\mathsf{T}} B_{\widehat{D}}^{\mathsf{T}} y && \text{by (10.6) and (23.8)} \\
&= y^{\mathsf{T}} B_{\widehat{D}} e_{(a,-\alpha)} && \text{by (10.20)} \\
&= y^{\mathsf{T}} \big(-\alpha B_D e_a\big) && \text{by (20.40)} \\
&= -y^{\mathsf{T}} B_{\widehat{D}} e_{(a,\alpha)} && \text{by (20.40)}
\end{aligned}$$

$$\begin{aligned} &= -e_{(a,\alpha)}^{\mathsf{T}} B_{\widehat{D}}^{\mathsf{T}} y && \text{by (10.20)} \\ &= -(\widehat{B}^{\mathsf{T}} y)_{(a,\alpha)} && \text{by (10.6) and (23.8).} \end{aligned}$$

Together with (23.14), this proves (23.22b).

Let C be a cycle in $\widehat{D}$. Then

$$\begin{aligned} (\widehat{B}^{\mathsf{T}} y)(C) &= \mathbb{1}_C^{\mathsf{T}} (\widehat{B}^{\mathsf{T}} y) && \text{by (11.12)} \\ &= y^{\mathsf{T}} \widehat{B} \mathbb{1}_C && \text{by (10.20)} \\ &= 0 && \text{by (11.14).} \end{aligned}$$

This proves (23.22c).

Each item in (23.22) specializes to the corresponding item in (23.23) by taking $y := y_t$, using the definition (23.19).

The inequality in (23.24) follows from (23.23a), using (23.18) and the inequality constraint in ($\mathrm{M}\mu\mathrm{C}_t$). The equality case follows from Proposition 23.2. □

23.6 Theorem (Monotonicity of Mean Costs). Restore the context from §23.4. If $t \in T$ is such that $t+1 \in T$, then

$$\varepsilon_{t+1} \leq \varepsilon_t \tag{23.25}$$

Proof. Since $t+1 \in T$, we have $t \in T'$. Thus, by (23.17),

$$\varepsilon_t > 0. \tag{23.26}$$

We will prove that

$$(y_t, \varepsilon_t) \text{ is feasible for } (\mathrm{M}\mu\mathrm{C}_{t+1}). \tag{23.27}$$

From this it will follow that optimal value of ($\mathrm{M}\mu\mathrm{C}_{t+1}$), which is ε_{t+1} by (23.12), is $\leq \varepsilon_t$.

By the definition of the LP in ($\mathrm{M}\mu\mathrm{C}_{t+1}$), proving (23.27) reduces to proving that

$$k_{t+1} - B_{t+1}^{\mathsf{T}} y_t \geq -\varepsilon_t \mathbb{1}. \tag{23.28}$$

By (23.13) applied to $t+1$, the vector inequality (23.28) is equivalent to one (real number) inequality for each arc $b \in A_{t+1}$.

Let $b \in A_{t+1}$. By (23.20), we have

$$b \in A_t \cup \overleftarrow{A(C_t)}. \tag{23.29}$$

If $b \in A_t$, then

$$\begin{aligned} (k_{t+1} - B_{t+1}^{\mathsf{T}} y_t)(b) &= (\widehat{k} - \widehat{B}^{\mathsf{T}} y_t)(b) && \text{by (23.22a)} \\ &= \widehat{k}_t(b) && \text{by (23.19)} \\ &\geq -\varepsilon_t && \text{by (23.24).} \end{aligned} \tag{23.30}$$

If $b \in \overleftarrow{A(C_t)}$, then by (20.43)

$$b = (a, -\alpha) \tag{23.31}$$

for some

$$(a, \alpha) \in A(C_t). \tag{23.32}$$

Thus,

$$\begin{aligned} (k_{t+1} - B_{t+1}^{\mathsf{T}} y_t)(b) &= (\widehat{k} - \widehat{B}^{\mathsf{T}} y_t)\big((a, -\alpha)\big) && \text{by (23.22a) and (23.31)} \\ &= \widehat{k}_t\big((a, -\alpha)\big) && \text{by (23.19)} \\ &= -\widehat{k}_t\big((a, \alpha)\big) && \text{by (23.23b)} \\ &= \varepsilon_t && \text{by (23.24) and (23.32)} \\ &> -\varepsilon_t && \text{by (23.26).} \end{aligned} \tag{23.33}$$

From (23.29), (23.30), and (23.33), it follows that (23.28) holds, thus completing the proof of (23.27). □

23.7 Lemma (Combinatorial Progress, Part One). Restore the context from §23.4. If $t \in T$ is such that $t+m \in T'$, then there is an arc $b \in \bigcup_{t \le \tau < t+m} A(C_\tau)$ such that $\widehat{k}_t(b) \ge 0$.

Proof. Suppose, for the sake of contradiction, that

$$\widehat{k}_t\big((a,\alpha)\big) < 0 \qquad \text{for each integer } \tau \in [t, t+m) \text{ and each } (a,\alpha) \in A(C_\tau). \tag{23.34}$$

Define

$$\widetilde{A}_\tau \coloneqq \big\{\, b \in A_\tau : \widehat{k}_t(b) < 0 \big\} \qquad \text{for each integer } \tau \in [t, t+m]. \tag{23.35}$$

We will prove that

$$\widetilde{A}_{\tau+1} \subsetneq \widetilde{A}_\tau \qquad \text{for each integer } \tau \in [t, t+m). \tag{23.36}$$

Let $\tau \in [t, t+m)$ be an integer, so that $\tau \in T'$. We start by proving the inclusion

$$\widetilde{A}_{\tau+1} \subseteq \widetilde{A}_\tau. \tag{23.37}$$

Let

$$b \in \widetilde{A}_{\tau+1}, \tag{23.38}$$

so that

$$b \in A_{\tau+1}, \tag{23.39}$$

$$\widehat{k}_t(b) < 0. \tag{23.40}$$

Thus, by (23.39) and (23.20),

$$b \in A_\tau \cup \overleftarrow{A(C_\tau)}. \tag{23.41}$$

We claim that

$$b \in A_\tau. \tag{23.42}$$

Suppose otherwise, for the sake of contradiction. Then, by (23.41),

$$b = (a, -\alpha) \tag{23.43}$$

for some

$$(a,\alpha) \in A(C_\tau), \tag{23.44}$$

Thus,

$$\begin{aligned} \widehat{k}_t(b) &= \widehat{k}_t\big((a,-\alpha)\big) && \text{by (23.43)} \\ &= -\widehat{k}_t\big((a,\alpha)\big) && \text{by (23.23b)} \\ &> 0 && \text{by (23.34) and (23.44).} \end{aligned}$$

This contradicts (23.40), and thus proves (23.42). From (23.42), (23.40), and (23.35), we get $b \in \widetilde{A}_\tau$. Since b in (23.38) was arbitrary, this proves (23.37).

By (23.21), there is

$$b \in A(C_\tau) \subseteq A_\tau \tag{23.45}$$

such that

$$b \notin A_{\tau+1}. \tag{23.46}$$

From (23.45) and (23.34) we get $\widehat{k}_t(b) < 0$ whence $b \in \widetilde{A}_\tau$ by (23.35). Furthermore, (23.46) and (23.35) show that $b \notin \widetilde{A}_{\tau+1}$. Hence, the inclusion in (23.37) is strict. This concludes the proof of (23.36).

By (23.23b), for each $a \in A$, at most one of $(a,+1)$ and $(a,-1)$ lies in $\widetilde{A}_t$. Thus, $|\widetilde{A}_t| \le m$. Together with (23.36), we find that $\widetilde{A}_{t+m} = \varnothing$. Thus,

$$\widehat{k}_t(b) \ge 0 \qquad \text{for each } b \in A_{t+m}. \tag{23.47}$$

Then

$$0 > -\varepsilon_{t+m}|C_{t+m}| \qquad \text{by (23.17), since } t+m \in T'$$

$$\begin{aligned}
&= \mu(D_{t+m}, k_{t+m})|C_{t+m}| && \text{by (23.9)}\\
&= k_{t+m}(C_{t+m}) && \text{by (23.16)}\\
&= \widehat{k}(C_{t+m}) && \text{by (23.14)}\\
&= \widehat{k}_t(C_{t+m}) && \text{by (23.23c)}\\
&\geq 0 && \text{by (23.47)},
\end{aligned}$$

a contradiction. □

Lecture 24: The Goldberg-Tarjan Bound

24.1 Theorem (A Dent in the Progress). Restore the context from §23.4. If $t \in T$ is such that $t + m \in T'$, then

$$\varepsilon_{t+m} \leq \Big(1 - \frac{1}{n}\Big)\varepsilon_t. \tag{24.1}$$

Proof. By Lemma 23.7, there is an ordered pair $(h, (a, \alpha))$ where

$$h \in [t, t + m) \tag{24.2}$$

is an integer and the arc

$$(a, \alpha) \in A(C_h) \tag{24.3}$$

satisfies

$$\widehat{k}_t\big((a, \alpha)\big) \geq 0. \tag{24.4}$$

Choose such an ordered pair that minimizes h. In particular,

$$\widehat{k}_t(b') < 0 \qquad \text{for each integer } \tau \in [t, h) \text{ and } b' \in A(C_\tau). \tag{24.5}$$

We will show that

$$\widehat{k}_t\big(b\big) \geq -\varepsilon_t \qquad \text{for each } b \in A(C_h). \tag{24.6}$$

Let $b \in A(C_h) \subseteq A_h$. By a straightforward induction using (23.41), we have

$$A_h \subseteq A_t \cup \bigcup_{t \leq \tau < h} \overleftarrow{A(C_\tau)}. \tag{24.7}$$

If $b \in A_t$, then (24.6) follows from (23.24). Otherwise, $b \in \overleftarrow{A(C_\tau)}$ for some integer

$$\tau \in [t, h). \tag{24.8}$$

Then

$$b = (c, -\gamma) \tag{24.9}$$

for some

$$(c, \gamma) \in A(C_\tau). \tag{24.10}$$

Then

$$\begin{aligned}
\widehat{k}_t(b) &= \widehat{k}_t\big((c, -\gamma)\big) && \text{by (24.9)}\\
&= -\widehat{k}_t\big((c, \gamma)\big) && \text{by (23.23b)}\\
&> 0 && \text{by (24.5), (24.8), and (24.10)}\\
&> -\varepsilon_t && \text{by (23.17)}.
\end{aligned}$$

This proves (24.6).

Hence,

$$\begin{aligned} k_h(C_h) &= \widehat{k}(C_h) && \text{by (23.14)} \\ &= \widehat{k}_t(C_h) && \text{by (23.23c)} \\ &= \widehat{k}_t\big((a,\alpha)\big) + \sum_{\substack{b\in A(C_h)\\ b\neq(a,\alpha)}} \widehat{k}_t(b) && \text{by (24.3)} \\ &\geq \sum_{\substack{b\in A(C_h)\\ b\neq(a,\alpha)}} \widehat{k}_t(b) && \text{by (24.4)} \\ &\geq -\varepsilon_t\big(|C_h|-1\big) && \text{by (24.6).} \end{aligned} \tag{24.11}$$

(Note the "dent" obtained above in the use of (24.4); that dent was obtained from the arduous combinatorial progress from Lemma 23.7.) Thus,

$$\frac{k_h(C_h)}{|C_h|} \geq -\varepsilon_t\Big(1-\frac{1}{|C_h|}\Big). \tag{24.12}$$

Finally,

$$\begin{aligned} -\varepsilon_{t+m} &\geq -\varepsilon_h && \text{by Theorem 23.6 and (24.2)} \\ &= \frac{k_h(C_h)}{|C_h|} && \text{by (23.9) and (23.16)} \\ &\geq -\varepsilon_t\Big(1-\frac{1}{|C_h|}\Big) && \text{by (24.12)} \\ &\geq -\varepsilon_t\Big(1-\frac{1}{n}\Big) && \text{since } |C_h|\leq n \text{ and } \varepsilon_t>0 \text{ by (23.17).} \end{aligned}$$

This concludes the proof. □

24.2 Exercise. Prove that[i]

$$1+x\leq e^x \qquad \text{for each } x\in\mathbb{R}, \tag{24.13}$$

with equality if and only if $x=0$.

24.3 Corollary (Boosting the Dent). Restore the context from §23.4. Set $\Delta t \coloneqq 2nm\lceil \ln n\rceil$. If $t\in T$ is such that $t+\Delta t\in T'$, then

$$\varepsilon_{t+\Delta t} < \frac{1}{2n}\varepsilon_t. \tag{24.14}$$

Proof. Theorem 24.1 and a simple induction show that

$$\varepsilon_{t+\Delta t} \leq \Big(1-\frac{1}{n}\Big)^{2n\lceil \ln n\rceil}\varepsilon_t. \tag{24.15}$$

Then

$$\begin{aligned} \Big(1-\frac{1}{n}\Big)^{2n\lceil \ln n\rceil} &< \big(e^{-1/n}\big)^{2n\lceil \ln n\rceil} && \text{by Exercise 24.2} \\ &= e^{-2\lceil \ln n\rceil} \\ &\leq e^{-2\ln n} \\ &= n^{-2} \\ &\leq \frac{1}{2n} && \text{since } n\geq 2 \text{ by hypothesis.} \end{aligned}$$

This concludes the proof. □

[i] By the Fundamental Theorem of Calculus, we have

$$\int_0^x (e^y-1)\,\mathrm{d}y = e^x-(1+x) \qquad \text{for each } x\in\mathbb{R}.$$

What can be said about the sign of the integrand (i.e., $y\mapsto e^y-1$) on the range of integration, in relation to the sign of x?

24.4 Theorem (Combinatorial Progress, Part Deux). Restore the context from §23.4. Set $\Delta t \coloneqq 2nm\lceil \ln n \rceil$. If $t \in T$ is such that $h \coloneqq t + \Delta t \in T'$, then there is an arc $(b,\beta) \in A(C_t)$ such that, for every $\tau \in T'$ such that $\tau \geq h$, we have $f_\tau(b) = f_h(b)$.

Proof. From $h \in T'$ and (23.17) we obtain

$$\varepsilon_h > 0. \tag{24.16}$$

We have

$$\begin{aligned} \widehat{k}_h(C_t) &= \widehat{k}(C_t) && \text{by (23.23c)} \\ &= k_t(C_t) && \text{by (23.14)} \\ &= -\varepsilon_t |C_t| && \text{by (23.9) and (23.16)}, \end{aligned}$$

whence there is $(b,\beta) \in A(C_t)$ such that $\widehat{k}_h\big((b,\beta)\big) \leq -\varepsilon_t$. Hence,

$$\begin{aligned} \widehat{k}_h\big((b,\beta)\big) &\leq -\varepsilon_t \\ &< -2n\varepsilon_h && \text{by Corollary 24.3} \\ &< -\varepsilon_h && \text{by (24.16)}. \end{aligned} \tag{24.17}$$

From (24.17), we get $\widehat{k}_h\big((b,\beta)\big) < -\varepsilon_h$, which shows via (23.24) that

$$(b,\beta) \notin A_h. \tag{24.18}$$

Suppose for the sake of contradiction that there is $\tau \in T'$ such that $\tau \geq h$ and

$$f_\tau(b) \neq f_h(b). \tag{24.19}$$

From $\tau \in T'$ and (23.17) we obtain

$$\varepsilon_\tau > 0. \tag{24.20}$$

Define

$$d\big((a,\alpha)\big) \coloneqq \alpha\big[(f_h - f_\tau)(a)\big]_\alpha \qquad \text{for each } (a,\alpha) \in \widehat{A}. \tag{24.21}$$

By Theorem 21.8,

$$d \text{ is a circulation in } \widehat{D}, \tag{24.22}$$

$$\operatorname{supp}(d) \subseteq A_\tau, \tag{24.23}$$

$$\overleftarrow{\operatorname{supp}(d)} \subseteq A_h. \tag{24.24}$$

Let us prove that

$$(b,\beta) \in \operatorname{supp}(d). \tag{24.25}$$

If not, (24.19) shows that $(b,-\beta) \in \operatorname{supp}(d)$. Thus,

$$(b,\beta) \in \overleftarrow{\operatorname{supp}(d)} \subseteq A_h, \tag{24.26}$$

where the last inclusion comes from (24.24). However, (24.26) contradicts (24.18), and thus proves (24.25).

Let $\mathcal{C}$ denote the set of cycles of the digraph $(V, \operatorname{supp}(d), \varphi\restriction_{\operatorname{supp}(d)})$. Theorem 18.7 shows that there is $y\colon \mathcal{C} \to \mathbb{R}_+$ such that

$$d = \sum_{C \in \mathcal{C}} y_C \mathbb{1}_C, \tag{24.27}$$

Specializing to the (b,β)th entry of (24.27), we obtain

$$\begin{aligned} 0 < d\big((b,\beta)\big) &&& \text{by (24.25) and (24.22)} \\ &= e_{(b,\beta)}^{\mathsf{T}} d &&\text{by (10.6)} \\ &= e_{(b,\beta)}^{\mathsf{T}} \Big(\sum_{C \in \mathcal{C}} y_C \mathbb{1}_C \Big) &&\text{by (24.27)} \\ &= \sum_{C \in \mathcal{C}} y_C e_{(b,\beta)}^{\mathsf{T}} \mathbb{1}_C &&\text{by (10.17c)} \end{aligned}$$

$$= \sum_{C \in \mathcal{C}} y_C \big[(b,\beta) \in A(C)\big].$$

Hence, there is $C \in \mathcal{C}$ such that $(b,\beta) \in A(C)$. That is,

$$\begin{aligned} (b,\beta) &\in A(C) \\ &\subseteq \operatorname{supp}(d) && \text{since } C \in \mathcal{C} \\ &\subseteq A_\tau && \text{by (24.23).} \end{aligned} \tag{24.28}$$

Thus,

$$C \text{ is a cycle in } D_\tau. \tag{24.29}$$

We will prove that

$$\frac{k_\tau(C)}{|C|} < \mu(D_\tau, k_\tau), \tag{24.30}$$

which will be a contradiction, and will thus complete the proof.

As an intermediate step, let us show that

$$\widehat{k}_h\big((a,\alpha)\big) \le \varepsilon_h \qquad \text{for each } (a,\alpha) \in A(C). \tag{24.31}$$

Let $(a,\alpha) \in A(C)$. Then

$$\begin{aligned} (a,-\alpha) &\in \overleftarrow{A(C)} \\ &\subseteq \overleftarrow{\operatorname{supp}(d)} && \text{since } A(C) \subseteq \operatorname{supp}(d) \text{ by (24.28)} \\ &\subseteq A_h && \text{by (24.24).} \end{aligned}$$

Thus,

$$\widehat{k}_h\big((a,-\alpha)\big) \ge -\varepsilon_h \tag{24.32}$$

by (23.24), and so

$$\begin{aligned} \widehat{k}_h\big((a,\alpha)\big) &= -\widehat{k}_h\big((a,-\alpha)\big) && \text{by (23.23b)} \\ &\le \varepsilon_h && \text{by (24.32).} \end{aligned}$$

This proves (24.31).

Finally,

$$\begin{aligned} k_\tau(C) &= \widehat{k}(C) && \text{by (23.14)} \\ &= \widehat{k}_h(C) && \text{by (23.23c)} \\ &= \widehat{k}_h\big((b,\beta)\big) + \sum_{\substack{(a,\alpha)\in A(C)\\(a,\alpha)\neq(b,\beta)}} \widehat{k}_h\big((a,\alpha)\big) && \text{by (24.28)} \\ &< -2n\varepsilon_h + \sum_{\substack{(a,\alpha)\in A(C)\\(a,\alpha)\neq(b,\beta)}} \widehat{k}_h\big((a,\alpha)\big) && \text{by (24.17)} \\ &\le -2n\varepsilon_h + \big(|C|-1\big)\varepsilon_h && \text{by (24.31)} \\ &= -\varepsilon_h\big(2n - |C| + 1\big) \\ &\le -\varepsilon_h\big(n+1\big) && \text{by (24.16), since } |C| \le n \\ &< -n\varepsilon_h && \text{by (24.16)} \\ &\le -n\varepsilon_\tau && \text{by Theorem 23.6, since } \tau \ge h \\ &\le -|C|\varepsilon_\tau && \text{by (24.20), since } |C| \le n. \end{aligned}$$

Dividing through by $|C|$ yields (24.30), via (23.9). □

24.5 Exercise. Restore the context from §23.4. Prove[1] that Algorithm 23.1 performs at most $4nm^2\lceil \ln n \rceil + O(1)$ iterations, and thus runs in polynomial time.

24.6 Exercise. Explain the following statement: when Algorithm 23.1 is applied to the reduction from the Maximum Flow Problem described in §18.12, the execution of the algorithm is equivalent to the execution of the Edmonds-Karp Algorithm 15.2.

Lecture 25: Eureka, You Shrink!

25.1 Deletion of Vertices and Edges. Let $G \coloneqq (V, E, \varphi)$ be a graph. Let $S \subseteq V$ and $F \subseteq E$. Define

$$G - S \coloneqq G[V \setminus S];$$

this is the graph obtained by the deletion of (the vertices of) S from G. Similarly, define

$$G - F \coloneqq (V, E \setminus F, \varphi\restriction_{E\setminus F});$$

this is the graph obtained by the deletion of (the edges of) F from G.

25.2 Even and Odd Components. Let $G \coloneqq (V, E, \varphi)$ be a graph. A component[i] C of G is **even** if $|C|$ is even, and C is **odd** if $|C|$ is odd. Denote the set of odd components of G by $\mathrm{Odd}(G)$, and set

$$\mathrm{odd}(G) \coloneqq |\mathrm{Odd}(G)|,$$

the number of odd components of G.

25.3 Exercise. Let $G \coloneqq (V, E, \varphi)$ be a graph. Prove that, if $M \subseteq E$ is a matching, then

$$B_G \mathbb{1}_M = \mathbb{1}_{V_M}. \tag{25.1}$$

25.4 Theorem. Let $G \coloneqq (V, E, \varphi)$ be a graph. Let $M \subseteq E$ be a matching of G, and let $S \subseteq V$. For each graph H and vertex $v \in V(H)$, let $\kappa_H(v)$ denote the component of H containing v. Define

$$N \coloneqq \{\, f \in M : \varphi(f) \cap S \neq \varnothing \}. \tag{25.2}$$

Then

$$2|M| \leq |V| + |S| - \mathrm{odd}(G - S), \tag{25.3}$$

and equality holds in (25.3) if and only if

(25.4)
- (i) $S \subseteq V_M$,
- (ii) $M \cap E_\varphi[S] = \varnothing$, and
- (iii) the restriction of κ_{G-S} to the set of $(M \setminus N)$-free vertices is a bijection to $\mathrm{Odd}(G - S)$.

Proof. It may help to glance at Figure 25.1 to follow the proof.

By (25.2), we have

$$|\varphi(f) \cap S| \in \{1, 2\} \qquad \text{for each } f \in N. \tag{25.5}$$

Note that $M \setminus N$ is a matching in $G - S$. Let $\mathcal{C}$ denote the set of components of $G - S$, and define

$$M_C \coloneqq M \cap E_\varphi[C] \qquad \text{for each } C \in \mathcal{C}; \tag{25.6}$$

again, for each $C \in \mathcal{C}$,

$$M_C \text{ is a matching of the graph } C. \tag{25.7}$$

It is simple to verify that

$$M = N \cup \bigcup_{C \in \mathcal{C}} M_C$$

and that $\{N\} \cup \{\, M_C : C \in \mathcal{C}\}$ is disjoint, whence

$$|M| = |N| + \sum_{C \in \mathcal{C}} |M_C|. \tag{25.8}$$

[1] Theorem 24.4 shows that, after Δt iterations, the flow through some arc gets fixed, i.e., it is never changed again.

[i] Recall our identification convention from §6.11.

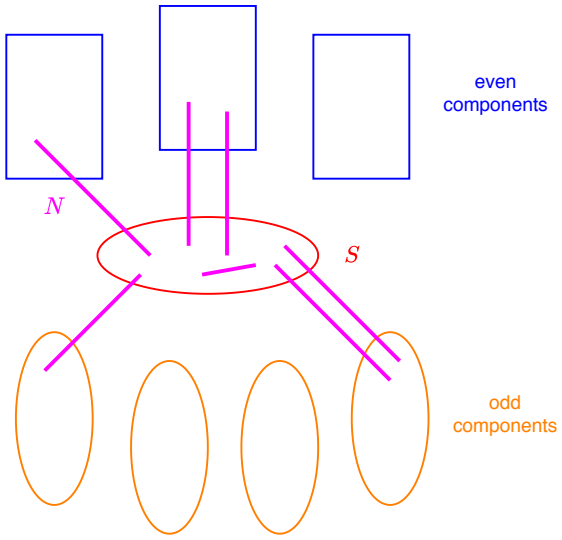


FIGURE 25.1. Schematic for Theorem 25.4.

We first prove that

$$|N| \leq |S|, \tag{25.9}$$

and that

(25.10) equality holds in (25.9) if and only if $S \subseteq V_N$ and $N \cap E_\varphi[S] = \varnothing$.

We shall bound the number $\mathbb{1}_S^\intercal B_G \mathbb{1}_N$ from below and from above.

We claim that

$$\mathbb{1}_S^\intercal B_G \mathbb{1}_N \geq |N| \tag{25.11}$$

and that

(25.12) equality holds in (25.11) if and only if $\big|S \cap \varphi(f)\big| = 1$ for each $f \in N$.

We have

$$\begin{aligned}
\mathbb{1}_S^\intercal B_G \mathbb{1}_N &= \mathbb{1}_S^\intercal B_G \sum_{f \in N} e_f && \text{by (10.9)} \\
&= \sum_{f \in N} \mathbb{1}_S^\intercal B_G e_f && \text{by (10.17c) and induction} \\
&= \sum_{f \in N} \mathbb{1}_S^\intercal \mathbb{1}_{\varphi(f)} && \text{by (10.28a)} \\
&= \sum_{f \in N} \big|S \cap \varphi(f)\big| && \text{by (10.11)} \\
&\geq \sum_{f \in N} 1 && \text{by (25.5), with equality if and only if } \big|S \cap \varphi(f)\big| = 1 \text{ for each } f \in N \\
&= |N|.
\end{aligned}$$

This proves (25.11) and (25.12).

Next, we show that

$$\mathbb{1}_S^\intercal B_G \mathbb{1}_N \leq |S| \tag{25.13}$$

and that

(25.14) equality holds in (25.13) if and only if $S \subseteq V_N$.

Indeed,

$$\begin{aligned} \mathbb{1}_S^\intercal B_G \mathbb{1}_N &= \mathbb{1}_S^\intercal \mathbb{1}_{V_N} && \text{by Exercise 25.3} \\ &= |S \cap V_N| && \text{by (10.11)} \\ &\leq |S| && \text{since } S \cap V_N \subseteq S, \text{ with equality if and only if } S \subseteq V_N. \end{aligned}$$

This proves (25.13) and (25.14).

Thus, by (25.11) and (25.13), we have $|N| \leq \mathbb{1}_S^\intercal B_G \mathbb{1}_N \leq |S|$, i.e., (25.9) holds, with equality throughout if and only if (25.12) and (25.14), which yields (25.10), since

$$N \cap E_\varphi[S] = \Big\{ f \in N : \big|S \cap \varphi(f)\big| = 2 \Big\}. \tag{25.15}$$

(Also, recall (25.5).)

For each $C \in \mathcal{C}$, we have by (25.7) that

$$|M_C| \leq \left\lfloor \frac{|C|}{2} \right\rfloor = \frac{|C| - [C \text{ is odd}]}{2}, \tag{25.16}$$

with equality if and only if the number of M_C-free vertices in C is precisely $[C \text{ is odd}]$. Thus,

$$\begin{aligned} 2 \sum_{C \in \mathcal{C}} |M_C| &\leq \sum_{C \in \mathcal{C}} \Big(|C| - [C \text{ is odd}] \Big) && \text{by (25.16)} \\ &= \sum_{C \in \mathcal{C}} |C| - \sum_{C \in \mathcal{C}} [C \text{ is odd}] \\ &= \big(|V| - |S| \big) - \operatorname{odd}(G - S), \end{aligned} \tag{25.17}$$

and equality holds if and only if (25.4)(iii) holds.

Thus,

$$\begin{aligned} 2|M| &= 2|N| + 2 \sum_{C \in \mathcal{C}} |M_C| && \text{by (25.8)} \\ &\leq 2|S| + \big(|V| - |S| \big) - \operatorname{odd}(G - S) && \text{by (25.9) and (25.16)} \\ &= |V| + |S| - \operatorname{odd}(G - S), \end{aligned}$$

with equality if and only if the condition from (25.10) and (25.4)(iii) hold. □

25.5 Maximum Matchings and Barriers. Let $G \coloneqq (V, E, \varphi)$ be a graph. Let $M \subseteq E$ be a matching of G, and let $S \subseteq V$. As a simple corollary to Theorem 25.4, if M is a matching and $S \subseteq V$ satisfy (25.3) with equality, then M is a maximum matching in G. In this case, the set S is called a **barrier** (of G).

25.6 Vertex Identification. Let $G \coloneqq (V, E, \varphi)$ be a graph. Let $S \subseteq V$. Denote

$$v/S \coloneqq \begin{cases} (1, v) & \text{if } v \notin S, \\ (2, S) & \text{if } v \in S, \end{cases} \qquad \text{for each } v \in V. \tag{25.18}$$

Define the graph $G/S \coloneqq \big((V \setminus S) \sqcup \{S\}, E, \varphi' \big)$, where

$$\varphi'(e) \coloneqq \big\{ v/S : v \in \varphi(e) \big\} \qquad \text{for each } e \in E.$$

The graph G/S is sometimes referred to as having been obtained from G by the identification of (the vertices in) S.

25.7 Matchings and Odd Cycles. Let $G \coloneqq (V, E, \varphi)$ be a graph. Let C be an odd cycle in G. Note that,

$$\text{for each } v \in V(C), \text{ the graph } C - v \text{ has a perfect matching.} \tag{25.19}$$

If $M \subseteq E$ is a matching, we say that C is **M-tight** if

$$\big| M \cap E(C) \big| = \left\lfloor \frac{|V(C)|}{2} \right\rfloor, \text{ and} \tag{25.20a}$$

$$|V(C) \setminus V_M| = 1. \tag{25.20b}$$

See Figures 25.2 and 25.3.

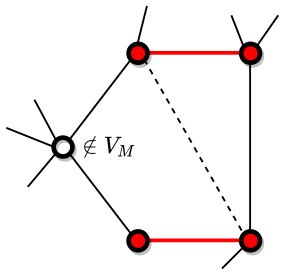
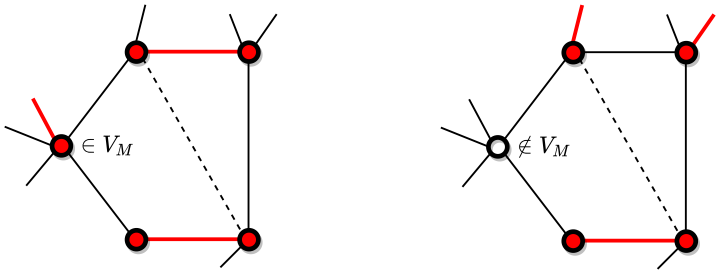


FIGURE 25.2. An example of an odd cycle that is M-tight. Note that the leftmost edge is M-free. The dashed edge is a chord of the odd cycle, i.e., it is an edge of the graph induced by the vertex set of the cycle, but not an edge of the cycle itself.



(A) (B)

FIGURE 25.3. Examples of odd cycles that are *not* M-tight. Th odd cycle in Figure 25.3a is not M-tight because $V(C) \setminus V_M = \varnothing$. The cycle of length 5 in Figure 25.3b is not M-tight because $M \cap E(C)$ does not have $\lfloor 5/2 \rfloor = 2$ edges.

25.8 Exercise (Eureka, You Shrink[i]). Let $G \coloneqq (V, E, \varphi)$ be a graph. Let C be an odd cycle in G. Set $U \coloneqq V(C)$ and $G' \coloneqq (G - E_\varphi[U])/U$.

(25.21) (i) If $M \subseteq E$ is a matching such that C is M-tight, then $M' \coloneqq M \setminus E_\varphi[U]$ is a matching in G' of size $|M'| = |M| - \lfloor |U|/2 \rfloor$ and the vertex $(2, U)$ from G' is M'-free.

(ii) If N' is a matching in G', v is any vertex of C if $(2, U)$ is N'-free or the unique vertex of C in $\bigcup_{e \in N'} \varphi(e)$ otherwise, and N_C is a perfect matching of $C - v$, then $N \coloneqq N' \cup N_C$ is a matching in G of size $|N| = |N'| + \lfloor |U|/2 \rfloor$.

25.9 Lemma (Expanding Certificates). Restore the context from (25.21)(i) from Exercise 25.8. If $S' \subseteq V' \coloneqq V(G')$ is such that

$$2|M'| = |V'| + |S'| - \operatorname{odd}(G' - S'), \tag{25.22}$$

then $(2, U) \notin S'$, and

$$S \coloneqq \{ v \in V : (1, v) \in S' \} \tag{25.23}$$

satisfies

$$2|M| = |V| + |S| - \operatorname{odd}(G - S). \tag{25.24}$$

[i] In reference to [6]. The seminal paper "Paths, trees, and flowers" [4] by Edmonds, which presents the algorithm from this lecture, is arguably one of the most influential in combinatorial optimization and computer science. It is widely cited as establishing, for the first time, the notion of polynomial-time solvability equated to algorithmic efficiency.

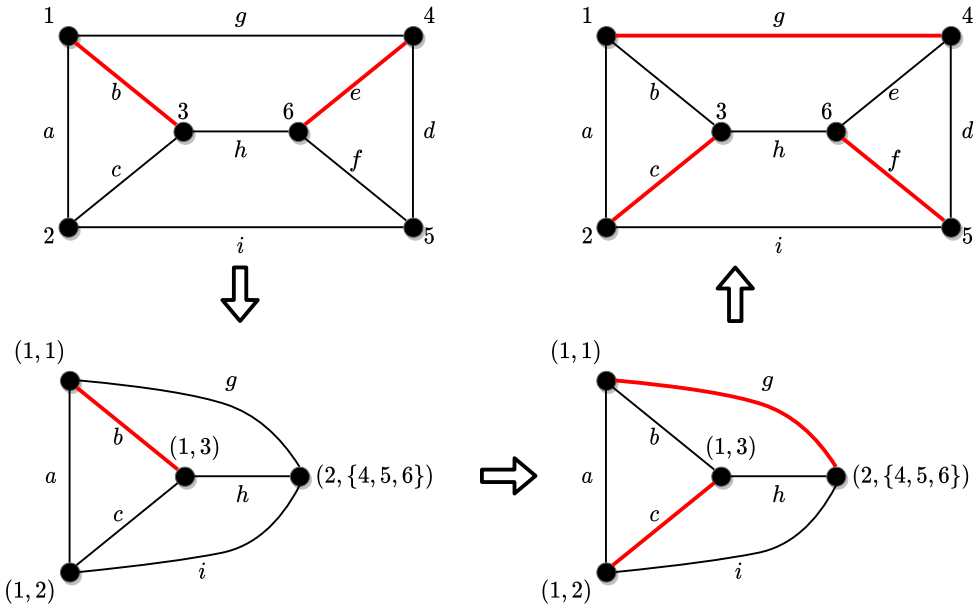


FIGURE 26.1. The procedure of shrinking an odd cycle, finding a larger matching, and expanding back to the original graph.

Proof Sketch. By (25.21)(i), the vertex $(2, U)$ is M'-free. Thus, by (25.22) and by the equality condition (25.4)(i) from Theorem 25.4, we find that $(2, U) \notin S'$. Verify as an exercise that[1]

$$\operatorname{odd}(G' - S') = \operatorname{odd}(G - S) \tag{25.25}$$

Thus

$$\begin{aligned}
2|M| &= 2|M'| + 2\lfloor |U|/2 \rfloor && \text{by (25.21)(i)} \\
&= |V'| + |S'| - \operatorname{odd}(G' - S') + 2\lfloor |U|/2 \rfloor && \text{by (25.22)} \\
&= |V'| + |S| - \operatorname{odd}(G - S) + 2\lfloor |U|/2 \rfloor && \text{by (25.23) and (25.25)} \\
&= |V'| + \big(|U| - 1\big) + |S| - \operatorname{odd}(G - S) && \text{since } |U| \text{ is odd} \\
&= |V| + |S| - \operatorname{odd}(G - S).
\end{aligned}$$

This proves (25.24). □

LECTURE 26: THE EDMONDS-PAP BLOSSOM ALGORITHM

26.1 An Augment-or-Certify Algorithm. Exercise 25.8 and Lemma 25.9 provide a recursive approach for an algorithm that takes as input a graph $G := (V, E, \varphi)$ and a matching $M \subseteq E$ of G, and finds either a larger matching, or a subset $S \subseteq V$ such that M and S satisfy (25.3) with equality. The approach is described as Algorithm 26.1[i], which relies on a procedure AUGMENT-OR-CERTIFY-OR-OBSTRUCT which we will design later. See Figure 26.1.

For now, it suffices to know that AUGMENT-OR-CERTIFY-OR-OBSTRUCT is a polynomial-time algorithm that takes as input an ordered pair (G, M), where G is a graph and M is a matching of G, and returns one of the following alternatives:

[1] There is a simple bijection between the set of components of $G' - S'$ and the set of components of $G - S$. For each component C' of $G' - S'$ not containing the vertex $(2, U)$, the correspondence is obvious (it is just a matter of "renaming" the vertices of the component), and the corresponding component of $G - S$ has the same parity as C'. The only "real difficulty" comes from the component of $G' - S'$ containing the vertex $(2, U)$. However, when one "expands" the vertex $(2, U)$ into U, the parity of the component is preserved, since $|U|$ is odd.

[i] This is not exactly Edmonds' algorithm, as presented in [4]. It incorporates modifications from [8, Sec. 1.3] to make it presentation more functional/local.

(i) a triple $(N, \perp, \perp)$, where N is a matching of G larger than M; or
(ii) a triple $(\perp, S, \perp)$, where $S \subseteq V$ satisfies $2|M| = |V| + |S| - \text{odd}(G - S)$, or
(iii) a triple $(N, \perp, C)$, where N is a matching of G such that $|N| = |M|$ and C is an N-tight odd cycle of G.

Algorithm 26.1 Augment-or-Certify Algorithm for Matchings

Input: a tuple (G, M), formed by
(i) a graph $G =: (V, E, \varphi)$,
(ii) a matching $M \subseteq E$.
Output: either one of the following:

(26.1) (i) $(N, \perp)$, where N is a matching of G such that $|N| > |M|$, or
(ii) $(\perp, S)$, where $S \subseteq V$ is such that $2|M| = |V| + |S| - \text{odd}(G - S)$.

```
1: function AUGMENT-OR-CERTIFY(G, M)
2:     (N, S, C) ← AUGMENT-OR-CERTIFY-OR-OBSTRUCT(G, M)
3:     if S ≠ ⊥ then                                   ▷ 2|M| = |V| + |S| − odd(G − S)
4:         return (⊥, S)
5:     else if C = ⊥ then                              ▷ N is a matching of G with |N| > |M|
6:         return (N, ⊥)
7:     else                  ▷ N is a matching of size |N| = |M| and C is an N-tight odd cycle in G
8:         M ← N
9:         U ← V(C)
10:        G' ← (G − E_φ[U])/U                          ▷ G' defined as in Exercise 25.8
11:        q ← (2, U)
12:        M' ← M \ E_φ[U]          ▷ M' is a matching of G' of size |M'| = |M| − ⌊|U|/2⌋ by (25.21)(i)
13:        (N', S') ← AUGMENT-OR-CERTIFY(G', M')
14:        if N' ≠ ⊥ then                               ▷ N' is a matching of G' with |N'| > |M'|
15:            if q is N'-free then
16:                Let v be any vertex of C
17:            else
18:                Let f ∈ N' having q as an end
19:                Let v ∈ U ∩ φ(f)
20:            Let N_C be a perfect matching of C − v                          ▷ see (25.19)
21:            return (N' ∪ N_C, ⊥) ▷ a matching of G of size |N'| + ⌊|U|/2⌋ > |M'| + ⌊|U|/2⌋ by (25.21)(ii)
22:        else
23:            Z ← { v ∈ V : (1, v) ∈ S' }
24:            return (⊥, Z)                  ▷ 2|M| = |V| + |Z| − odd(G − Z) by Lemma 25.9
```

26.2 Algorithm 26.1 Runs in Polynomial Time. The running time of Algorithm 26.1 is polynomial on the size of the input. Indeed, recall that AUGMENT-OR-CERTIFY-OR-OBSTRUCT runs in polynomial time. Upon executing AUGMENT-OR-CERTIFY on an input (G, M), a recursive call may be run in Line 13. However, the recursion arguments (G', M') are such that $|E(G')| < |E(G)|$, since C has at least one edge. Each recursive call either completes the task of finding a larger matching or a barrier and returns, in which case the call stack unwinds until the initial function call to AUGMENT-OR-CERTIFY returns, or else another recursive call is made. However, but the previous argument, the recursion depth is bounded by m, the number of edges of G. Hence, the running time is polynomial.

26.3 Isolated Vertices. Let $G := (V, E, \varphi)$ be a graph. A vertex $v \in V$ is **isolated** if there is no $e \in E$ such that $v \in \varphi(e)$; in this case note in particular that there is no loop on v.

Denote the set of isolated vertices of G by $\text{Iso}(G)$, and set

$$\text{iso}(G) := |\text{Iso}(G)|, \tag{26.2}$$

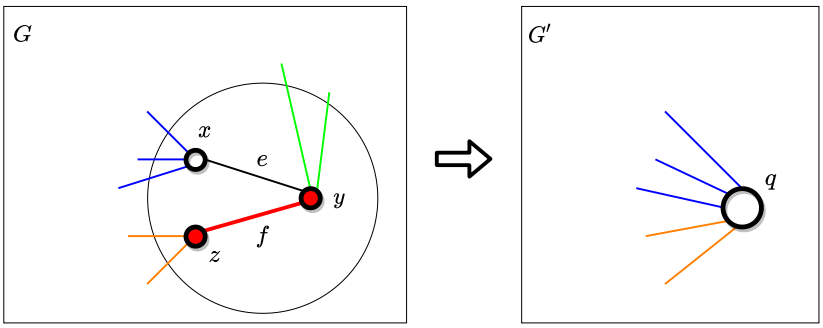


FIGURE 26.2

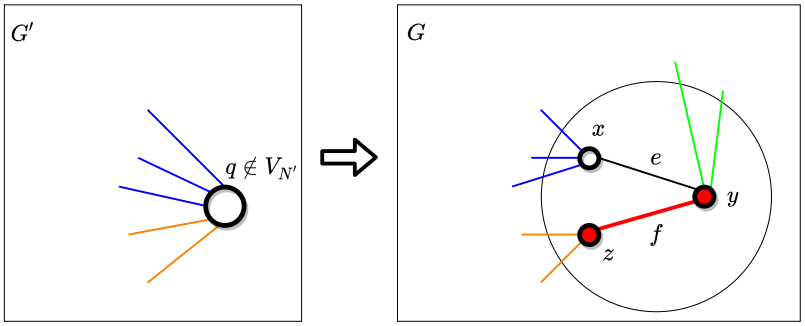


FIGURE 26.3

the number of isolated vertices of G. Clearly,

$$\operatorname{iso}(G) \leq \operatorname{odd}(G). \tag{26.3}$$

26.4 Proposition (Barriers of Isolated Vertices). Let $G \coloneqq (V, E, \varphi)$ be a graph. If $M \subseteq E$ is a matching of G and $S \subseteq V$ is such that

$$2|M| \geq |V| + |S| - \operatorname{iso}(G - S), \tag{26.4}$$

then M is a maximum matching and S is a barrier of G.

Proof. We have

$$\begin{aligned} 2|M| &\geq |V| + |S| - \operatorname{iso}(G - S), && \text{by (26.4)} \\ &\geq |V| + |S| - \operatorname{odd}(G - S), && \text{by (26.3)}, \end{aligned}$$

whence equality holds in (25.3). □

26.5 The Augment or Certify or Obstruct Algorithm. To complete the Edmonds-Pap algorithm for maximum matching, we must design algorithm AUGMENT-OR-CERTIFY-OR-OBSTRUCT as described in §26.1. The algorithm appears as Algorithm 26.2, with a slightly different/stronger interface. Namely, barriers satisfy the stronger condition described by Proposition 26.4. Correctness of the algorithm follows from the comments alongside the pseudocode; use induction/recursion. The running time is polynomial by arguments similar to those in §26.2.

LECTURE 27: THE TUTTE-BERGE FORMULA

27.1 Lemma. Let $G \coloneqq (V, E, \varphi)$ be a graph. Let $M \subseteq E$ be a matching. Let $x, y, z \in V$ be three distinct vertices, and suppose there are edges $e \in E \setminus M$ and $f \in M$ such that $\varphi(e) = xy$ and $\varphi(f) = yz$. Define the graph $G' \coloneqq (G - y)/\{x, z\}$ and the matching $M' \coloneqq M \setminus \{f\}$ in G'. Suppose $S' \subseteq V(G') =: V'$ is such that

$$2|M'| = |V'| + |S'| - \operatorname{iso}(G' - S'), \tag{27.1}$$

and set $S \coloneqq \{y\} \cup \{\, v \in V : (1, v) \in S' \,\}$. Then $2|M| = |V| + |S| - \operatorname{iso}(G - S)$.

Algorithm 26.2 Augment-or-Certify-or-Obstruct Algorithm for Matchings

Input: a tuple (G, M), formed by
(i) a graph $G =: (V, E, \varphi)$,
(ii) a matching $M \subseteq E$.
Output: either one of the following:

(26.5) (i) a triple $(N, \perp, \perp)$, where N is a matching of G of size $|M| + 1$; or
(ii) a triple $(\perp, S, \perp)$, where $S \subseteq V$ satisfies $2|M| = |V| + |S| - \text{iso}(G - S)$, or
(iii) a triple $(N, \perp, C)$, where N is a matching of G such that $|N| = |M|$ and C is an N-tight odd cycle of G.

1: **function** AUGMENT-OR-CERTIFY-OR-OBSTRUCT(G, M)
2: **if** $2|M| + \text{iso}(G) = |V|$ **then**
3: **return** $(\perp, \varnothing, \perp)$ ▷ $2|M| = |V| + |\varnothing| - \text{iso}(G - \varnothing)$
4: Let $x \in V$ such that $x \notin V_M \cup \text{Iso}(G)$ ▷ such vertex exists since the **if** from Line 2 failed
5: Let $e \in E \setminus M$ such that $x \in \varphi(e)$ ▷ such edge exists since $x \notin V_M \cup \text{Iso}(G)$
6: **if** e is a loop **then**
7: **return** $(M, \perp, \langle x, e, x\rangle)$ ▷ $\langle x, e, x\rangle$ is M-tight odd cycle
8: Let $y \in \varphi(e)$ such that $y \neq x$ ▷ such vertex exists since the **if** from Line 6 failed
9: **if** y is M-exposed **then**
10: **return** $(M \cup \{e\}, \perp, \perp)$ ▷ $M \cup \{e\}$ is a matching of size $|M| + 1$
11: Let $f \in M$ such that $y \in \varphi(f)$ ▷ such edge exists since the **if** from Line 9 failed
12: Let $z \in \varphi(f)$ such that $z \neq y$ ▷ such vertex exists since $f \in M$ is not a loop; $x \neq z \in V_M$
13: $G' \coloneqq (G - y)/\{x, z\}$ ▷ G' has two fewer vertices than G since $x \neq z$
14: $M' \coloneqq M \setminus \{f\}$ ▷ M' is a matching in G' of size $|M'| = |M| - 1$
15: $q \coloneqq (2, \{x, z\})$ ▷ abbreviate name of "big" vertex; q is M'-exposed; see Figure 26.2
16: $(N', S', C') \leftarrow$ AUGMENT-OR-CERTIFY-OR-OBSTRUCT(G', M')
17: **if** $S' \neq \perp$ **then** ▷ $2|M'| = |V(G')| + |S'| - \text{iso}(G' - S')$
18: **return** $(\perp, S, \perp)$ where $S \coloneqq \{y\} \cup \{\, v \in V : (1, v) \in S'\}$ ▷ see Lemma 27.1
19: **else** ▷ N' is a matching in G that exposes y and at least one of x or z
20: **if** z is N'-exposed **then**
21: $N \leftarrow N' \cup \{f\}$ ▷ N is a matching in G with $|N| = |N'| + 1$
22: **else** ▷ x is N'-exposed since the **if** from Line 20 failed; see comment from Line 19
23: $N \leftarrow N' \cup \{e\}$ ▷ N is a matching in G with $|N| = |N'| + 1$
24: **if** $C' = \perp$ **then** ▷ N' is a matching in G' with $|N'| = |M'| + 1$; see Figures 26.3 to 26.5
25: **return** $(N, \perp, \perp)$ ▷ N is a matching in G with $|N| = |N'| + 1 = |M'| + 2 = |M| + 1$
26: **else** ▷ N' is a matching in G' with $|N'| = |M'|$ and C' is an N'-tight odd cycle in G'
27: $\langle (\alpha_0, v_0), \ldots, e_\ell, (\alpha_\ell, v_\ell)\rangle \leftarrow C'$, where $(\alpha_0, v_0) = (\alpha_\ell, v_\ell) = q = (2, \{x, z\})$ if $q \in V(C')$
28: **if** $q \notin V(C')$ **then**
29: $C \leftarrow \langle v_0, \ldots, e_\ell, v_\ell\rangle$ ▷ C is an N-tight odd cycle in G; verify this!
30: **return** $(N, \perp, C)$ ▷ matching N in G has size $|N| = |N'| + 1 = |M'| + 1 = |M|$
31: **else** ▷ $(\alpha_0, v_0) = (\alpha_\ell, v_\ell) = q = (2, \{x, z\})$ and $\alpha_i = 1$ for each $i \in [\ell - 1]$
32: Let u_0 be the end of e_1 in $\{x, z\}$
33: Let u_ℓ be the end of e_ℓ in $\{x, z\}$
34: $C_0 \leftarrow \langle u_0, e_1, v_1, \ldots, e_{\ell-1}, v_{\ell-1}, e_\ell, u_\ell\rangle$ ▷ C_0 is a walk in G with $|E(C_0) \cap N'| = \lfloor \ell/2 \rfloor$
35: **if** $u_0 = x$ and $u_\ell = x$ **then** ▷ C_0 is a cycle through x; see Figures 26.6 and 26.7
36: **return** $(N' \cup \{f\}, \perp, C_0)$ ▷ C_0 is $(N' \cup \{f\})$-tight
37: **else if** $u_0 = z$ and $u_\ell = z$ **then** ▷ C_0 is a cycle through z
38: **return** $(N' \cup \{e\}, \perp, C_0)$ ▷ C_0 is $(N' \cup \{e\})$-tight; verify!
39: **else if** $u_0 = x$ and $u_\ell = z$ **then** ▷ "broken" cycle upon expansion; see Figures 26.8 and 26.9
40: **return** $(N, \perp, C_0 \cdot \langle z, f, y, e, x\rangle)$ ▷ odd cycle $C_0 \cdot \langle z, f, y, e, x\rangle$ in G is N-tight; verify!
41: **else if** $u_0 = z$ and $u_\ell = x$ **then** ▷ mirror image of Line 39
42: **return** $(N, \perp, C_0 \cdot \langle x, e, y, f, z\rangle)$

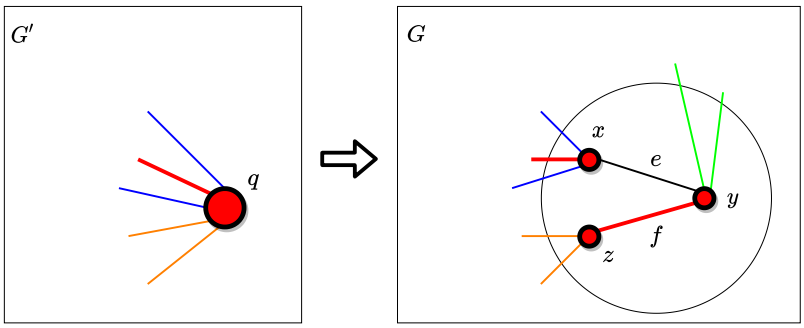


FIGURE 26.4

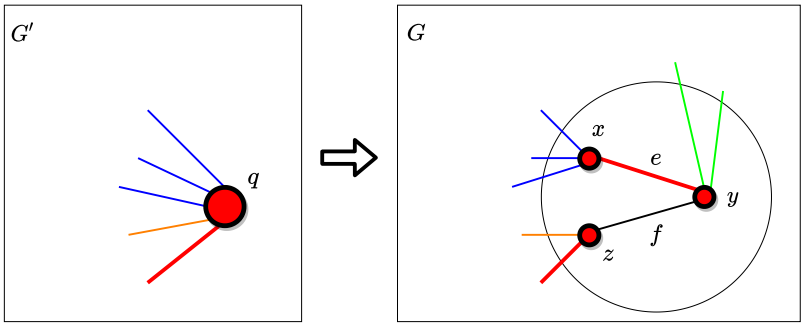


FIGURE 26.5

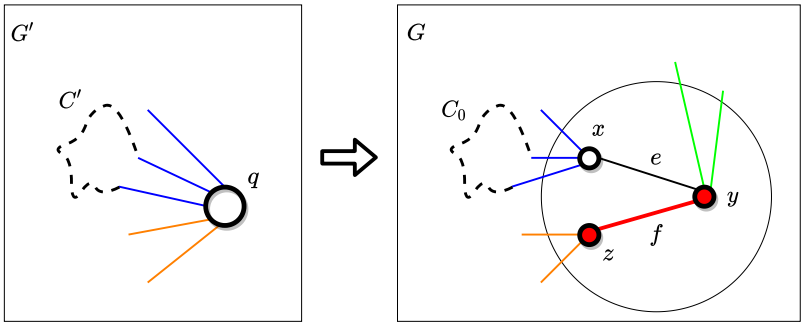


FIGURE 26.6

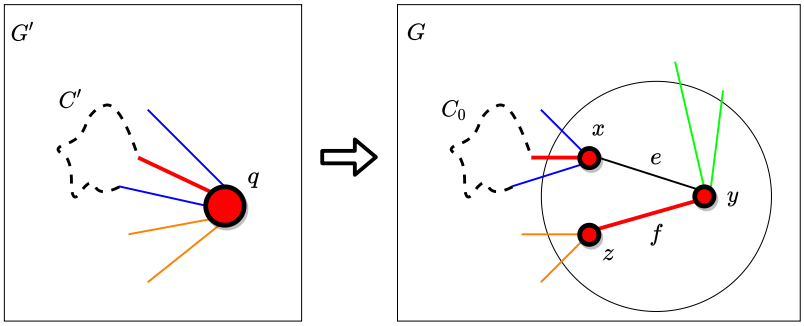


FIGURE 26.7

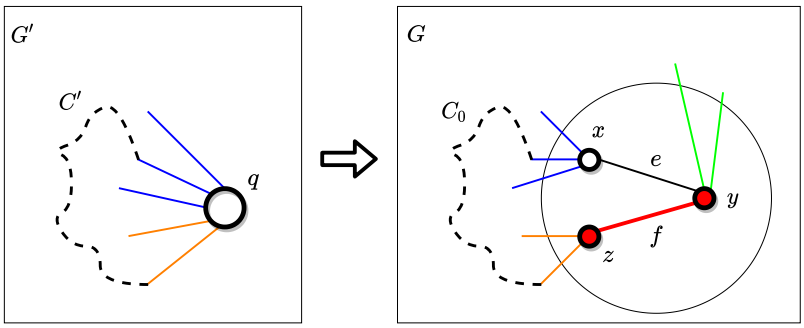


FIGURE 26.8

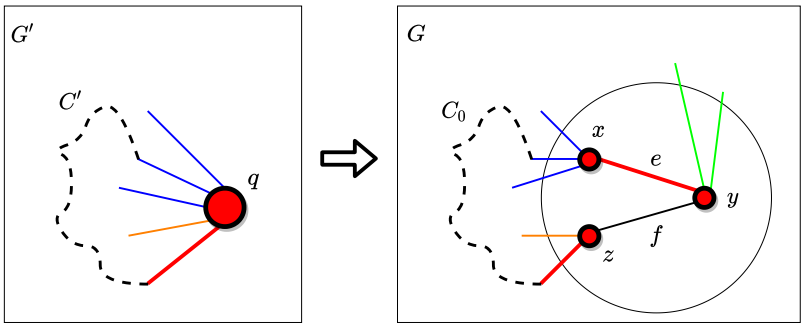


FIGURE 26.9

Proof. Unpack $G' =: (V', E', \varphi')$. By (27.1) and Proposition 26.4, we find that equality holds in (25.3). Hence, by items (25.4)(i) and (25.4)(ii),

$$S' \text{ is a subset of the set } V_{M'} \text{ of } M'\text{-saturated vertices of } G', \tag{27.2}$$

and

$$M' \cap E'_{\varphi'}[S'] = \varnothing. \tag{27.3}$$

Since

$$q \coloneqq (2, \{x, z\}) \text{ is } M'\text{-exposed in } G', \tag{27.4}$$

we find from (27.2) that

$$q \notin S'. \tag{27.5}$$

The set $N' \coloneqq \{\, e \in M' : \varphi'(e) \cap S' = \varnothing\}$ is a matching in $G' - S'$. By (27.3), for each edge $e \in M'$ one has $|\varphi'(e) \cap S'|$ equal to 0 or 1, and exactly $|S'|$ of these are equal to 1 by (27.2). Hence,

$$|N'| = |M'| - |S'|. \tag{27.6}$$

We have

$$\begin{aligned} \big|V(G' - S')\big| &= |V'| - |S'| \\ &= 2|M'| - 2|S'| + \operatorname{iso}(G' - S') && \text{by (27.1)} \\ &= 2\big(|M'| - |S'|\big) + |\operatorname{Iso}(G' - S')| && \text{by (26.2)} \\ &= 2|N'| + |\operatorname{Iso}(G' - S')| && \text{by (27.6)} \\ &= |V_{N'}| + |\operatorname{Iso}(G' - S')|. \end{aligned} \tag{27.7}$$

Since the sets $V_{N'}$ of N'-saturated vertices of $G' - S'$ and $\operatorname{Iso}(G' - S')$ are disjoint, (27.7) shows that every vertex of $G' - S'$ that is N'-exposed lies in $\operatorname{Iso}(G' - S')$, i.e., is isolated in $G' - S'$. Thus, (27.4) shows that

$$q \in \operatorname{Iso}(G' - S'). \tag{27.8}$$

It is now a simple exercise to verify that

$$\mathrm{Iso}(G-S) = \big\{\, v \in V : (1,v) \in \mathrm{Iso}(G'-S') \big\} \cup \{x,z\} \tag{27.9}$$

and thus

$$\mathrm{iso}(G-S) = \mathrm{iso}(G'-S') + 1. \tag{27.10}$$

Hence,

$$\begin{aligned} 2|M| &= 2|M'| + 2 \\ &= |V'| + |S'| - \mathrm{iso}(G'-S') + 2 && \text{by (27.1)} \\ &= \big(|V|-2\big) + \big(|S|-1\big) - \big(\mathrm{iso}(G-S)-1\big) + 2 && \text{by (27.10)} \\ &= |V| + |S| - \mathrm{iso}(G-S). \end{aligned}$$

This concludes the proof. □

27.2 Theorem (The Tutte-Berge Formula). Let $G \coloneqq (V,E,\varphi)$ be a graph. Then

$$2\nu(G) = \min_{S \subseteq V} \Big(|V| + |S| - \mathrm{odd}(G-S) \Big). \tag{27.11}$$

Proof. Inequality '$\leq$' in (27.11) follows from Theorem 25.4. Let $M^* \subseteq E$ be a maximum matching of G, i.e., an optimal solution for (7.1); see (7.2). Feed (G, M^*) as input to Algorithm 26.1, and let (N, Z) be its return value. If $N \neq \perp$, then N is a matching larger than M^*, a contradiction. Thus, $N = \perp$ and Z is a subset of V such that $2|M^*| = |V| + |Z| - \mathrm{odd}(G-Z)$. This proves that

$$\begin{aligned} 2\nu(G) &= 2|M^*| \\ &= |V| + |Z| - \mathrm{odd}(G-Z) \\ &\geq \min_{S \subseteq V} \Big(|V| + |S| - \mathrm{odd}(G-S) \Big). \end{aligned}$$

This proves '$\geq$' in (27.11). □

27.3 Corollary (Tutte's Matching Theorem). Let $G \coloneqq (V,E,\varphi)$ be a graph. Then G has a perfect matching if and only if

$$\mathrm{odd}(G-S) \leq |S| \qquad \text{for each } S \subseteq V. \tag{27.12}$$

Proof. We have

$$\begin{aligned} G \text{ has a perfect matching} &\iff 2\nu(G) = |V| \\ &\iff |V| = \min_{S \subseteq V} \Big(|V| + |S| - \mathrm{odd}(G-S) \Big) \quad \text{by Theorem 27.2} \\ &\iff 0 = \min_{S \subseteq V} \Big(|S| - \mathrm{odd}(G-S) \Big). \end{aligned} \tag{27.13}$$

Suppose G has a perfect matching. Let $S \subseteq V$. By (27.13), we have $|S| - \mathrm{odd}(G-S) \geq 0$, i.e., (27.12) holds.

Conversely, suppose (27.12) holds. Then

$$\min_{S \subseteq V} \Big(|S| - \mathrm{odd}(G-S) \Big) \geq 0 \tag{27.14}$$

and

$$\mathrm{odd}(G) \leq 0; \tag{27.15}$$

the latter is obtained by applying (27.12) with $S \coloneqq \varnothing$. It follows from (27.15) that $\mathrm{odd}(G) = 0$, so $|\varnothing| - \mathrm{odd}(G-\varnothing) = 0$, which shows that (27.14) holds with equality. □

27.4 Matchings and LPs. In *all* optimization problems we have solved (provided efficient algorithms for) so far, we had an LP relaxation of an IP and we proved that one can always find an optimal solution for the LP relaxation which is integral, provided the data is integral. In other words, the LP relaxation is *exact*. We may consider this phenomenon for the Maximum Matching Problem 7.2 and the IP (11.7), along with its LP relaxation (11.8).

Let $G \coloneqq (V, E, \varphi)$ be an odd cycle. It is simple to verify that $\frac{1}{2}\mathbb{1}$ is feasible in (11.8) with objective value $|V|/2$. Yet, it is clear (due to parity) that $\nu(G) = \lfloor |V|/2 \rfloor < |V|/2$. Since the IP (11.7) is isomorphic to the maximum matching optimization problem (7.1), it follows that

(27.16) the LP relaxation (11.8) has no optimal solution that is integral

for this graph.

27.5 Exercise (Integrality Gap). Prove that, for every $M \in \mathbb{N}$, there is a graph G such that the optimal value of the LP relaxation (11.8) is $\geq \nu(G) + M$.

27.6 Valid Inequalities for Matchings. Let $G \coloneqq (V, E, \varphi)$ be a loopless graph. In order to prevent the catastrophe from Exercise 27.5, one may add inequality constraints to the LP relaxation (11.8) that hold for each vector of the form $\mathbb{1}_M$, where $M \subseteq E$ is a matching. Note that such inequalities are *not necessary* for the IP formulation (11.7), which is isomorphic to the maximum matching optimization problem (7.1).

Let $M \subseteq E$ be a matching of G. Let $U \subseteq V$ such that $|U|$ is odd. Then $M \cap E_\varphi[U]$ is a matching of $G[U]$, whence

$$\big|M \cap E_\varphi[U]\big| \leq \left\lfloor \frac{|U|}{2} \right\rfloor. \tag{27.17}$$

Thus, the incidence vector $x \coloneqq \mathbb{1}_M$ satisfies

$$\begin{aligned} x\big(E_\varphi[U]\big) &= \mathbb{1}_{E_\varphi[U]}^\intercal x && \text{by (10.12)} \\ &= \mathbb{1}_{E_\varphi[U]}^\intercal \mathbb{1}_M \\ &= |M \cap E_\varphi[U]| && \text{by (10.11)} \\ &\leq \left\lfloor \frac{|U|}{2} \right\rfloor. \end{aligned}$$

27.7 The Matching LP. Let $G \coloneqq (V, E, \varphi)$ be a loopless graph. Define

$$\mathcal{P}_{\mathrm{odd}}(V) \coloneqq \big\{ U \subseteq V : |U| \text{ is odd} \big\}. \tag{27.18}$$

By §27.6, every feasible solution x for (11.7) is a feasible solution for the LP

$$\text{Maximize} \quad x(E) \tag{27.19a}$$

$$\text{subject to} \quad x\big(\delta(v)\big) \leq 1 \qquad \text{for each } v \in V, \tag{27.19b}$$

$$x\big(E_\varphi[U]\big) \leq \left\lfloor \frac{|U|}{2} \right\rfloor \qquad \text{for each } U \in \mathcal{P}_{\mathrm{odd}}(V), \tag{27.19c}$$

$$x \in \mathbb{R}_+^E. \tag{27.19d}$$

In particular,

(27.20) the optimal value of (27.19) is $\geq \nu(G)$.

The dual LP of (27.19) is

$$\text{Minimize} \quad y(V) + \sum_{U \in \mathcal{P}_{\mathrm{odd}}(V)} \left\lfloor \frac{|U|}{2} \right\rfloor z_U \tag{27.21a}$$

$$\text{subject to} \quad y \in \mathbb{R}_+^V, \tag{27.21b}$$

$$z \in \mathbb{R}_+^{\mathcal{P}_{\mathrm{odd}}(V)} \tag{27.21c}$$

$$y\big(\varphi(e)\big) + \sum \big\{ z_U : U \in \mathcal{P}_{\mathrm{odd}}(V),\, \varphi(e) \subseteq U \big\} \geq 1 \qquad \text{for each } e \in E. \tag{27.21d}$$

We will prove in Theorem 27.9 that there is a homomorphism from the optimization problem

$$\text{Minimize} \quad \tfrac{1}{2}\Big(|V| + |S| - \mathrm{odd}(G - S)\Big) \tag{27.22a}$$

$$\text{subject to} \quad S \subseteq V \tag{27.22b}$$

to (27.21). From this and (6.5) it will follow that

(27.23) the optimal value of (27.21) is $\leq \frac{1}{2} \min_{S \subseteq V} \Big(|V| + |S| - \mathrm{odd}(G - S)\Big)$.

Combine (27.23) with Theorem 27.2 to obtain

(27.24) the optimal value of (27.21) is $\leq \nu(G)$.

Finally, combine (27.20) and (27.24) and LP Weak Duality (see Theorem 11.2) to obtain that

(27.25) the optimal value of (27.19) is $\nu(G)$.

From this it follows that the LP (27.19) has an integral optimal solution.

27.8 An Algorithm. Let $G \coloneqq (V, E, \varphi)$ be a loopless graph. Let $S \subseteq V$. Algorithm 27.1 shows how to obtain a subset $Y \subseteq V$ such that

$$|Y| - \operatorname{odd}(G - Y) \leq |S| - \operatorname{odd}(G - S) \tag{27.26a}$$

and

(27.26b) every component of $G - Y$ is odd.

Consider an iteration of the **for** loop in Line 3. If $G - Y_t$ has no even component, then (27.26b) holds for Y_t; see Line 8. We claim that

(27.27) if Y_{t+1} is defined, then $|Y_{t+1}| - \operatorname{odd}(G - Y_{t+1}) \leq |Y_t| - \operatorname{odd}(G - Y_t)$.

Clearly, the number of iterations is bounded by $|V| + 1$. Thus, a simple induction shows that the subset returned satisfies (27.26a).

Let us prove (27.27). Set $H \coloneqq G - Y_t$. Let $C \coloneqq C_t$ be an even component of H, and let $v \coloneqq v_t \in C_t$. Then the components of $G - Y_{t+1} = H - v$ are all the components of H, except for C, together with all the components of $C - v$. Thus,

$$\operatorname{odd}(G - Y_{t+1}) = \operatorname{odd}(G - Y_t) + \operatorname{odd}(C - v). \tag{27.28}$$

Since $C - v$ has an odd number of vertices, at least one of its components is odd, so $\operatorname{odd}(C - v) \geq 1$. Hence, by (27.28),

$$\operatorname{odd}(G - Y_{t+1}) \geq \operatorname{odd}(G - Y_t) + 1, \tag{27.29}$$

whence

$$\begin{aligned} |Y_{t+1}| - \operatorname{odd}(G - Y_{t+1}) &= |Y_t| + 1 - \operatorname{odd}(G - Y_{t+1}) \\ &\leq |Y_t| + 1 - \big(\operatorname{odd}(G - Y_t) + 1\big) \quad \text{by (27.29)} \\ &= |Y_t| - \operatorname{odd}(G - Y_t). \end{aligned}$$

This proves (27.27).

Algorithm 27.1 Algorithm for Maximal Odd Cover

Input: a tuple (G, S), formed by
(i) a loopless graph $G = (V, E, \varphi)$,
(ii) a subset $S \subseteq V$.
Output: a subset $Y \subseteq V$ that satisfies the conditions (27.26).

```
1: function MAXIMAL-ODD-COVER(G, S)
2:     Y_0 ← S
3:     for t ← 0 to ∞ do
4:         if G − Y_t has an even component C_t then
5:             Let v_t ∈ V(C_t)
6:             Y_{t+1} ← Y_t ∪ {v_t}
7:         else
8:             return Y_t
```

27.9 Theorem. Let $G \coloneqq (V, E, \varphi)$ be a loopless graph. Let $S \subseteq V$. Let Y be returned by Algorithm 27.1 with input (G, S). Set

$$y \coloneqq \mathbb{1}_Y, \tag{27.30a}$$

$$z_U \coloneqq \big[U \text{ is a component of } G - Y\big] \qquad \text{for each } U \in \mathcal{P}_{\mathrm{odd}}(V). \tag{27.30b}$$

Then

$$(y, z) \text{ is feasible for (27.21)} \tag{27.31}$$

with objective value $\leq \frac{1}{2}\Big(|V| + |S| - \mathrm{odd}(G - S)\Big)$.

Proof. Clearly y and z are nonnegative, i.e., (27.21b) and (27.21c) are met. Let $e \in E$. If $\varphi(e) \cap Y \neq \varnothing$, then

$$y\big(\varphi(e)\big) + \sum\big\{\, z_U : U \in \mathcal{P}_{\mathrm{odd}}(V),\ \varphi(e) \subseteq U \big\} \geq y\big(\varphi(e)\big) = |Y \cap \varphi(e)| \geq 1.$$

Otherwise, e is an edge of $G - Y$, so there is a component C of $G - Y$ such that $\varphi(e) \subseteq C$. By (27.26b), C is odd, so $z_C = 1$. Thus,

$$y\big(\varphi(e)\big) + \sum\big\{\, z_U : U \in \mathcal{P}_{\mathrm{odd}}(V),\ \varphi(e) \subseteq U \big\} \geq z_C = 1.$$

This concludes the proof of (27.31).

Its objective value is

$$\begin{aligned}
y(V) + \sum_{U \in \mathcal{P}_{\mathrm{odd}}(V)} \Big\lfloor \frac{|U|}{2} \Big\rfloor z_U &= \mathbb{1}^{\mathsf{T}} y + \sum_{U \in \mathcal{P}_{\mathrm{odd}}(V)} \Big\lfloor \frac{|U|}{2} \Big\rfloor z_U && \text{by (10.12)} \\
&= \mathbb{1}^{\mathsf{T}} \mathbb{1}_Y + \sum_{U \in \mathrm{Odd}(G-Y)} \frac{|U| - 1}{2} && \text{by (27.30)} \\
&= |Y| + \frac{1}{2}\Big(\sum_{U \in \mathrm{Odd}(G-Y)} |U| - \sum_{U \in \mathrm{Odd}(G-Y)} 1 \Big) && \text{by (10.11)} \\
&= |Y| + \frac{1}{2}\Big(\big(|V| - |Y|\big) - \mathrm{odd}(G - Y) \Big) && \text{by (27.26b)} \\
&= \frac{1}{2}\Big(|V| + |Y| - \mathrm{odd}(G - Y) \Big) \\
&\leq \frac{1}{2}\Big(|V| + |S| - \mathrm{odd}(G - S) \Big) && \text{by (27.26a).}
\end{aligned}$$

This completes the proof. □

Lecture 28: Exam Prep Question Session

This lecture is reserved for answering questions from previous material, in preparation for the final exam.

Lecture 29: Minimum Spanning Trees

29.1 Problem (Maximum Weight Matchings). The **maximum-weight matching problem** is, **given**

(i) a graph $G = (V, E, \varphi)$, and
(ii) a weight function $w \colon E \to \mathbb{R}$,

solve the optimization problem

$$\begin{aligned} &\text{Maximize} && w(M) \\ &\text{subject to} && M \text{ is a matching in } G. \end{aligned} \tag{29.1}$$

Note that (29.1) generalizes the Maximum Matching Problem 7.2, obtained by taking $w \coloneqq \mathbb{1}$ in (29.1).

29.2 Exponential-Sized, Highly Structured LPs. As discussed in §27.7, Algorithm 26.1 solves the LP (27.19) and its dual (27.21), which have exponentially many constraints and variables, respectively. It can be shown that, if one replaces the objective function (27.19a) to $w^{\mathsf{T}} x$, the optimal value of the resulting LP

$$\text{Maximize} \quad w^{\mathsf{T}} x \tag{29.2a}$$

$$\text{subject to} \quad x\big(\delta(v)\big) \leq 1 \qquad \text{for each } v \in V, \tag{29.2b}$$

$$x\big(E_\varphi[U]\big) \leq \left\lfloor \frac{|U|}{2} \right\rfloor \qquad \text{for each } U \in \mathcal{P}_{\mathrm{odd}}(V), \tag{29.2c}$$

$$x \in \mathbb{R}^E_+, \tag{29.2d}$$

has an integral optimal solution, which shows that its optimal value is equal to the optimal value of (29.1).

Not only that, it is possible to combine the ideas from the primal-dual Hungarian method (Algorithm 19.2) and the Edmonds-Pap Blossom Algorithm (Algorithm 26.1) to design a polynomial-time algorithm for solving (29.1). Such an algorithm relies on the formulation (29.2) and its dual,

$$\text{Minimize} \quad y(V) + \sum_{U \in \mathcal{P}_{\mathrm{odd}}(V)} \left\lfloor \frac{|U|}{2} \right\rfloor z_U \tag{29.3a}$$

$$\text{subject to} \quad y \in \mathbb{R}^V_+, \tag{29.3b}$$

$$z \in \mathbb{R}^{\mathcal{P}_{\mathrm{odd}}(V)}_+, \tag{29.3c}$$

$$y\big(\varphi(e)\big) + \sum \big\{ z_U : U \in \mathcal{P}_{\mathrm{odd}}(V),\, \varphi(e) \subseteq U \big\} \geq w(e) \qquad \text{for each } e \in E. \tag{29.3d}$$

See [2, Sec. 5.3].

This can be achieved because one does not blindly solve the LP (29.2) by examining the exponentially many constraints for each feasible solution. Instead, feasibility in the primal LP (29.2) is preserved by keeping a matching M at each iteration, with a corresponding primal feasible solution $\mathbb{1}_M$ which necessarily satisfies all the constraints (29.2c); see §27.7. Similarly, in the dual, a current dual feasible solution can be maintained by keeping y and *only the elements of* $\mathrm{supp}(z)$, which may be kept polynomial-sized[i].

This idea provides a powerful extended interpretation of the Fundamental Paradigm of Combinatorial Optimization from §11.6. Namely, one starts with an IP formulation for the original combinatorial optimization problem at hand. Then one relaxes the IP to an LP, while simultaneously adding "valid inequalities" as in §27.6 so that the resulting LP has an integral optimal solution; a fundamental theorem due to Minkowski and Weyl shows that this is always possible. Then one can set up the dual LP and bring in the big guns based on complementary slackness conditions.

The discussion above shows that, even if the resulting (primal) LP has exponentially many constraints, provided the constraints are "well structured"[ii], it may be possible to solve the corresponding optimization problem. We will see next a complete example of that, using the well-known Kruskal's algorithm for minimum spanning trees.

29.3 Problem (Minimum Spanning Tree). Let $G \coloneqq (V, E, \varphi)$ be a graph. A **spanning tree** of G is a spanning subgraph of G that is a tree.

The **minimum(-cost) spanning tree problem** is, **given**

(i) a graph $G \coloneqq (V, E, \varphi)$, and
(ii) a cost function $c\colon E \to \mathbb{R}$,

solve the optimization problem

$$\begin{aligned} &\text{Minimize} && c(F) \\ &\text{subject to} && F \text{ is the edge set of a spanning tree of } G. \end{aligned} \tag{29.4}$$

29.4 Exercise (Equivalent Characterizations of Trees). Let $G \coloneqq (V, E, \varphi)$ be a graph. Prove that the following are equivalent:[iii]

(29.5)
(i) G is a tree;
(ii) G is connected and $|E| = |V| - 1$;
(iii) G is acyclic and $|E| = |V| - 1$.

[i] In the context of Integer Programming, one can think that each dual variable z_U is generated only when one detects that it is advantageous to do so; this is sometimes referred to as "column generation" in the dual LP.

[ii] This is necessarily a vague term.

[iii] A standard proof technique in graph theory involving trees is to rely on the fact that every tree with at least 2 vertices has a leaf, i.e., a vertex that has precisely *one* neighbor.

29.5 Exercise (The Number of Components Function). Let $G \coloneqq (V, E, \varphi)$ be a graph. Throughout this lecture and the next, let $\kappa\colon \mathcal{P}(E) \to \mathbb{N}$ denote the function defined by taking $\kappa(F)$ to be the number of components of the subgraph $(V, F, \varphi\restriction_F)$ of G, for each $F \subseteq E$. Prove that

$$\kappa(F') \geq \kappa(F) \qquad \text{whenever } F' \subseteq F \subseteq E. \tag{29.6}$$

29.6 Proposition (An LP Formulation for Minimum Spanning Trees). Let $G \coloneqq (V, E, \varphi)$ be a graph. Let $c\colon E \to \mathbb{R}$. Then the map $F \subseteq E \mapsto \mathbb{1}_F$ is a homomorphism from (29.4) to the LP

$$\begin{aligned}
&\text{Minimize} && c^{\mathsf{T}} x && && \text{(29.7a)}\\
&\text{subject to} && x(E) = |V| - 1, && && \text{(29.7b)}\\
& && x(F) \leq |V| - \kappa(F) && \text{for each } F \subseteq E, && \text{(29.7c)}\\
& && x \in \mathbb{R}^E_+. && && \text{(29.7d)}
\end{aligned}$$

Proof. Let T be a spanning tree of G. We first verify that

$$\bar{x} \coloneqq \mathbb{1}_{E(T)} \text{ is feasible for (29.7).} \tag{29.8}$$

Note that

$$\begin{aligned}
\bar{x}(E) &= \mathbb{1}_E^{\mathsf{T}} \bar{x} && \text{by (10.12)}\\
&= \mathbb{1}_E^{\mathsf{T}} \mathbb{1}_{E(T)} && \text{by (29.8)}\\
&= |E(T)| && \text{by (10.11)}\\
&= |V| - 1 && \text{by Exercise 29.5,}
\end{aligned}$$

so constraint (29.7b) is satisfied.

Let $F \subseteq E$, and set

$$A \coloneqq E(T) \cap F \subseteq F. \tag{29.9}$$

Set $\psi \coloneqq \varphi\restriction_A$. Since the graph $H \coloneqq (V, A, \psi)$ is a subgraph of the acyclic graph T, the graph H is acyclic. Let $\mathcal{C}$ denote the set of components of H. Then each component $C \in \mathcal{C}$ is a tree, so

$$|A_\psi[C]| = |C| - 1 \tag{29.10}$$

by Exercise 29.4. Since $A = \bigcup_{C \in \mathcal{C}} A_\psi[C]$ and $\{\, A_\psi[C] : C \in \mathcal{C} \}$ is (pairwise) disjoint, we find that

$$\begin{aligned}
|A| &= \sum_{C \in \mathcal{C}} |A_\psi[C]|\\
&= \sum_{C \in \mathcal{C}} \big(|C| - 1\big) && \text{by (29.10)}\\
&= |V| - \kappa(A).
\end{aligned} \tag{29.11}$$

Hence,

$$\begin{aligned}
\bar{x}(F) &= \mathbb{1}_F^{\mathsf{T}} \bar{x} && \text{by (10.12)}\\
&= \mathbb{1}_F^{\mathsf{T}} \mathbb{1}_{E(T)} && \text{by (29.8)}\\
&= |F \cap E(T)| && \text{by (10.11)}\\
&= |A| && \text{by (29.9)}\\
&= |V| - \kappa(A) && \text{by (29.11)}\\
&\leq |V| - \kappa(F) && \text{by (29.9) and Exercise 29.5.}
\end{aligned}$$

Thus, the constraints (29.7c) are satisfied. Since clearly $\bar{x} \geq 0$, this concludes the proof of (29.8).

The objective value of $\bar{x}$ in (29.7) is

$$c^{\mathsf{T}} \bar{x} = c^{\mathsf{T}} \mathbb{1}_{E(T)} = c(E(T)),$$

which is exactly the objective value of $E(T)$ in (29.4). □

29.7 The Dual LP. Let $G \coloneqq (V, E, \varphi)$ be a graph. Let $c\colon E \to \mathbb{R}$. The dual LP of (29.7) is

$$\text{Maximize} \quad \big(|V| - 1\big)\eta + \sum_{F \subsetneq E} \big(|V| - \kappa(F)\big) y_F \tag{29.12a}$$

$$\text{subject to} \quad \eta \in \mathbb{R}, \tag{29.12b}$$

$$y \in \mathbb{R}_-^{\mathcal{P}(E)}, \tag{29.12c}$$

$$\eta \mathbb{1} + \sum_{F \subsetneq E} y_F \mathbb{1}_F \leq c. \tag{29.12d}$$

This dual LP is feasible, regardless of G being connected or not. Indeed, verify that

$$\bar{\eta} \coloneqq \min_{e \in E} c(e) \text{ and } \bar{y} \coloneqq 0 \text{ yield a feasible solution for (29.12).} \tag{29.13}$$

29.8 Unbounded Dual LP for Disconnected Graphs. Restore the context from §29.7. Suppose that G is not connected. Then clearly (29.4) is infeasible. However, nothing *a priori* ensures that the LP (29.7) is infeasible. We will prove that this is the case, by showing that the dual LP (29.12) is unbounded; from this, it follows by LP Weak Duality that the primal LP (29.7) is infeasible, whence so is (29.4), by Proposition 29.6.

Define $\bar{\eta}$ and $\bar{y}$ as in (29.13), and set

$$\begin{aligned} \eta_\lambda &\coloneqq \bar{\eta} + \lambda \in \mathbb{R} \\ y_\lambda &\coloneqq \bar{y} - \lambda e_E \in \mathbb{R}^{\mathcal{P}(E)}, \end{aligned} \qquad \text{for each } \lambda \in \mathbb{R}_+. \tag{29.14}$$

Let $\lambda \in \mathbb{R}_+$. It is trivial that (29.12b) holds for η_λ, and (29.12c) holds for $y_\lambda \leq \bar{y}$ by (29.13). The LHS of (29.12d) applied to η_λ and y_λ is

$$\eta_\lambda \mathbb{1} + \sum_{F \subsetneq E} y_\lambda(F) \mathbb{1}_F = \bar{\eta}\mathbb{1} + \lambda \mathbb{1} + \sum_{F \subsetneq E} \bar{y}_F \mathbb{1}_F - \lambda \mathbb{1}_E = \bar{\eta}\mathbb{1} + \sum_{F \subsetneq E} \bar{y}_F \leq c, \tag{29.15}$$

using (29.13). Thus,

$$\eta_\lambda \text{ and } y_\lambda \text{ form a feasible solution for (29.12).}$$

If μ denotes the objective value of $(\bar{\eta}, \bar{y})$ in (29.12), the objective value of η_λ, y_λ is

$$\mu + \lambda\Big(\big(|V| - 1\big) - \big(|V| - \kappa(E)\big)\Big) = \mu + \lambda\big(\kappa(E) - 1\big). \tag{29.16}$$

Since $\kappa(E) > 1$, the objective value can be made arbitrarily large, by sending $\lambda \to \infty$.

29.9 Exercise. Let $G \coloneqq (V, E, \varphi)$ be a *connected* graph. Let $c\colon E \to \mathbb{R}$. Prove that the LP (29.7) and the LP

$$\text{Minimize} \quad c^\mathsf{T} x \tag{29.17a}$$

$$\text{subject to} \quad x(E) = |V| - 1, \tag{29.17b}$$

$$x(F) \leq |V| - \kappa(F) \qquad \text{for each } F \subsetneq E, \tag{29.17c}$$

$$x \in \mathbb{R}_+^E. \tag{29.17d}$$

are homomorphically equivalent through the identity function. Also, prove that the dual LP

$$\text{Maximize} \quad \sum_{F \subsetneq E} \big(|V| - \kappa(F)\big) y_F \tag{29.18a}$$

$$\text{subject to} \quad y_E \in \mathbb{R}, \tag{29.18b}$$

$$y_F \in \mathbb{R}_- \qquad \text{for each } F \subsetneq E, \tag{29.18c}$$

$$\sum_{F \subsetneq E} y_F \mathbb{1}_F \leq c. \tag{29.18d}$$

of (29.17) is homomorphically equivalent to the dual LP (29.12).

29.10 Complementary Slackness. Let $G \coloneqq (V, E, \varphi)$ be a connected graph. Let $c\colon E \to \mathbb{R}$. The complementary slackness conditions for the dual pair of LPs in Exercise 29.9 are,

$$\big(e \in E \text{ and } x(e) > 0\big) \implies \sum_{F \subsetneq E} \big[e \in F\big] y_F = c(e), \tag{29.19a}$$

and

$$\big(F \subsetneq E \text{ and } y_F < 0\big) \implies x(F) = |V| - \kappa(F). \tag{29.19b}$$

In other words, if $x \in \mathbb{R}^E$ and $y \in \mathbb{R}^{\mathcal{P}(E)}$ are vectors, then x and y are optimal solutions for (29.17) and (29.18), respectively, if and only if

(29.20) (i) x is feasible for (29.17),
(ii) y is feasible for (29.18),
(iii) the complementary slackness conditions from (29.19) hold for x and y.

For completeness, we prove the latter statement.

29.11 Proposition. Let $G \coloneqq (V, E, \varphi)$ be a connected graph. Let $c\colon E \to \mathbb{R}$. Let x be a feasible solution for (29.17) and let y be a feasible solution for (29.18). Denote

$$r(F) \coloneqq |V| - \kappa(F) \qquad \text{for each } F \subseteq E. \tag{29.21}$$

Then

$$c^{\mathsf{T}} x \geq r^{\mathsf{T}} y, \tag{29.22}$$

with equality if and only if (29.19) holds.

Proof. By applying (11.1)(iv) and using (29.17d) and (29.18d), we find that

$$c^{\mathsf{T}} x \geq \Big(\sum_{F \subseteq E} y_F \mathbb{1}_F\Big)^{\mathsf{T}} x \tag{29.23}$$

and

(29.24) (29.23) holds with equality if and only if (29.19a) holds.

By (10.12), the RHS of (29.23) is

$$\begin{aligned} \sum_{F \subseteq E} y_F x(F) &= y_E x(E) + \sum_{F \subsetneq E} y_F x(F) \\ &= y_E r(E) + \sum_{F \subsetneq E} (-y_F)\big(-x(F)\big) \quad \text{by (29.17b)} \\ &\geq y_E r(E) + \sum_{F \subsetneq E} (-y_F)\big(-r(F)\big), \quad \text{by (29.17c) and (29.18c)}, \end{aligned} \tag{29.25}$$

and

(29.26) (29.25) holds with equality if and only if (29.19b) holds.

Finally, the rightmost term in (29.25) is easily checked to be

$$\sum_{F \subseteq E} y_F r(F) = r^{\mathsf{T}} y.$$

Thus, by combining (29.25) we obtain (29.22), and the equality case (29.19) follows from (29.24) and (29.26). □

Lecture 30: The Greedy and the Dual Greedy Algorithms

30.1 The Greedy and the Dual Greedy Algorithms. Let $G \coloneqq (V, E, \varphi)$ be a connected graph with m edges. Let $c\colon E \to \mathbb{R}$. Define a bijection $f\colon [m] \to E$ so that[1]

$$c(f_1) \leq \cdots \leq c(f_m). \tag{30.1}$$

Define[2]

$$\begin{aligned} A_i &\coloneqq \{f_1, \ldots, f_i\}, \\ G_i &\coloneqq (V, A_i, \varphi\restriction_{A_i}), \end{aligned} \qquad \text{for each } i \in \{0, \ldots, m\}. \tag{30.2}$$

[1] This is the sorting part of Kruskal's algorithm.
[2] These are the sets of edges analyzed until a certain iteration of Kruskal's algorithm.

Define $x \in \mathbb{R}^E$ and $y \in \mathbb{R}^{\mathcal{P}(E)}$ by setting

$$x(f_i) \coloneqq \kappa(A_{i-1}) - \kappa(A_i) \qquad \text{for each } i \in [m], \tag{30.3}$$

$$y_F \coloneqq \sum_{i=1}^{m} [F = A_i]\Big(c(f_i) - [i < m]c(f_{i+1})\Big) \qquad \text{for each } F \subseteq E. \tag{30.4}$$

Let $i \in [m]$. Since A_i is obtained from A_{i-1} by adding the edge f_i, the graph G_i has either the same number of components of G_{i-1} or one fewer; more precisely,

$$x(f_i) = \big[\text{the ends of } f_i \text{ lie in distinct components of } G_{i-1}\big] \quad \text{for each } i \in [m]. \tag{30.5}$$

Thus,

$$x \in \{0,1\}^E. \tag{30.6}$$

Set

$$T \coloneqq (V, \operatorname{supp}(x), \varphi\restriction_{\operatorname{supp}(x)}), \tag{30.7}$$

so that

$$x = \mathbb{1}_{E(T)}; \tag{30.8}$$

see §18.4.

We will (eventually) prove that T is a spanning tree of G and that (29.20) holds. From this, it will follow that x and y are optimal solutions for (29.17) and (29.18), respectively, and that T is an optimal solution for (29.4), via Proposition 29.6.

30.2 Proposition. Restore the context from §30.1. Then T is a spanning tree of G.

Proof. Since T is clearly a spanning subgraph of G, it suffices to prove that T is a tree. We have

$$\begin{aligned}
|E(T)| &= \mathbb{1}^{\mathsf{T}} \mathbb{1}_{E(T)} && \text{by (10.11)} \\
&= \mathbb{1}^{\mathsf{T}} x && \text{by (30.8)} \\
&= \sum_{i=1}^{m} x(f_i) \\
&= \sum_{i=1}^{m} \big(\kappa(A_{i-1}) - \kappa(A_i)\big) && \text{by (30.3)} \\
&= \kappa(A_0) - \kappa(A_m) && \text{by telescoping} \\
&= \kappa(\varnothing) - \kappa(E) && \text{by (30.2)} \\
&= |V| - 1 && \text{since } G \text{ is connected.}
\end{aligned}$$

Hence, by Exercise 29.4, it suffices to prove that T is acyclic. We will prove by induction on i that[1]

$$\text{the graph } H_i \coloneqq (V, E(T) \cap A_i, \varphi\restriction_{E(T)\cap A_i}) \text{ is acyclic, for each } i \in \{0, \ldots, m\}. \tag{30.9}$$

For the base case $i = 0$, the graph has no edges, so (30.9) clearly holds. Let $i \in [m]$. By induction hypothesis, H_{i-1} is acyclic. Suppose for the sake of contradiction that H_i has a cycle C. Then $H_i \neq H_{i-1}$, whence

$$f_i \in E(T), \tag{30.10}$$

and f_i is the only element of $E(H_i) \setminus E(H_{i-1})$. Note that $f_i \in E(C)$; otherwise, C is a cycle in H_{i-1}, which was proved to be acyclic. Since $C - f_i$ is a path in H_{i-1} joining the ends of f_i,

$$\text{the ends of } f_i \text{ lie in the same component of } H_{i-1}. \tag{30.11}$$

Since H_{i-1} is a spanning subgraph of G_{i-1}, the Iverson in (30.5) evaluates to 0, so $x(f_i) = 0$ whence $f_i \notin \operatorname{supp}(x) = E(T)$; this contradicts (30.10) and concludes the proof of the inductive step of (30.9). □

[1] You may think of this statement and its proof as the invariant preserved by Kruskal's algorithm that the current forest is a maximal forest in the subgraph G_i.

30.3 Theorem. Restore the context from §30.1. Then y is feasible for (29.18), and in fact the inequality (29.18d) holds with equality.

Proof. By (30.4), the support of y is contained in $\{A_1, \ldots, A_m\}$. Let $i \in [m]$. The membership constraint (29.18b) on $y(A_m) = y(E)$ does not impose on its sign, so assume $i < m$. Then $y(A_i) = c(f_i) - c(f_{i+1}) \leq 0$ by (30.1). Thus, the membership constraints (29.18b) and (29.18c) are satisfied by y.

Let us prove that (29.18d) holds with equality. Let $j \in [m]$. Then

$$\begin{aligned}
\Big(\sum_{F \subseteq E} y_F \mathbb{1}_F\Big)(f_j) &= e_{f_j}^\mathsf{T}\Big(\sum_{F \subseteq E} y_F \mathbb{1}_F\Big) && \text{by (10.6)} \\
&= \sum_{F \subseteq E} y_F e_{f_j}^\mathsf{T} \mathbb{1}_F && \text{by (10.17c) and induction} \\
&= \sum_{F \subseteq E} y_F \big[f_j \in F\big] && \text{by (10.10)} \\
&= \sum_{F \subseteq E} \sum_{i=1}^{m} \big[F = A_i\big]\Big(c(f_i) - [i < m]c(f_{i+1})\Big)\big[f_j \in F\big] && \text{by (30.4)} \\
&= \sum_{i=1}^{m} \Big(c(f_i) - [i < m]c(f_{i+1})\Big)\big[f_j \in A_i\big] \\
&= \sum_{i=1}^{m} \Big(c(f_i) - [i < m]c(f_{i+1})\Big)[i \geq j] && \text{by (30.2)} \\
&= \sum_{i=j}^{m} \Big(c(f_i) - [i < m]c(f_{i+1})\Big) \\
&= c(f_m) + \sum_{i=j}^{m-1} \Big(c(f_i) - c(f_{i+1})\Big) \\
&= c(f_m) + \big(c(f_j) - c(f_m)\big) && \text{by telescoping} \\
&= c(f_j).
\end{aligned}$$

□

30.4 Corollary. Restore the context from §30.1. Then (29.20) holds for x and y.

Proof. Item (29.20)(i) follows from Propositions 29.6 and 30.2. Item (29.20)(ii) follows from Theorem 30.3. To prove (29.20)(iii), note that (29.19a) follows trivially from Theorem 30.3. It remains only to verify (29.19b). Let $F \subsetneq E$ such that $y_F < 0$. By (30.4), we must have

$$F = A_i \tag{30.12}$$

for some $i \in [m-1]$. Then

$$\begin{aligned}
x(F) &= x(A_i) && \text{by (30.12)} \\
&= \mathbb{1}_{A_i}^\mathsf{T} x && \text{by (10.12)} \\
&= \sum_{j=1}^{i} x(f_j) && \text{by (30.2)} \\
&= \sum_{j=1}^{i} \big(\kappa(A_{j-1}) - \kappa(A_j)\big) && \text{by (30.3)} \\
&= \kappa(A_0) - \kappa(A_i) && \text{by telescoping} \\
&= \kappa(\varnothing) - \kappa(A_i) && \text{by (30.2)} \\
&= |V| - \kappa(A_i).
\end{aligned}$$

□

30.5 Let $G \coloneqq (V, E, \varphi)$ be a graph. Let $c\colon E \to \mathbb{R}$. By the discussions in §29.8 and §30.1, the optimization problems (29.4) and (29.7) are homomorphically equivalent. One may ask whether the exponentially many inequalities in (29.17) are required. The answer is negative: many inequalities in (29.7) may be derived from other inequalities. Consider the (much smaller) LP

$$\begin{aligned}
&\text{Minimize} && c^{\mathsf{T}}x && && \text{(30.13a)}\\
&\text{subject to} && x(E) = |V| - 1, && && \text{(30.13b)}\\
& && x\big(E_\varphi[U]\big) \le |U| - 1 && \text{for each nonempty } U \subseteq V, && \text{(30.13c)}\\
& && x \in \mathbb{R}^E_+. && && \text{(30.13d)}
\end{aligned}$$

We will show that

(30.14) a vector $x \in \mathbb{R}^E$ is feasible for (29.7) if and only if x is feasible for (30.13).

Note, however, that (30.13) still has exponentially many constraints, and one can prove (see [9, Sec. 50.4]) that (homomorphic) equivalence between (29.4) and (30.13) for every cost function $c \in \mathbb{R}^E$ requires every inequality of the form (30.13c) such $|U| \ge 2$ and $G[U]$ is 2-connected[i].

Proof of (30.14). Let x be feasible for (29.7). Note that (30.13b) and (29.7b) are identical, and so are (30.13d) and (29.7d). Thus, to prove that x is feasible for (30.13), it suffices to prove that (30.13c) holds. Let $U \subseteq V$ be nonempty. Set

$$F \coloneqq E_\varphi[U]. \tag{30.15}$$

Then the components of the graph $(V, F, \varphi\restriction_F)$ are formed by isolated vertices in $\overline{U} \coloneqq V \setminus U$ and the components of $G[U]$. Since $U \neq \varnothing$, there is at least one component in $G[U]$. Hence,

$$\kappa(F) \ge |\overline{U}| + 1 = |V| - |U| + 1. \tag{30.16}$$

Thus,

$$\begin{aligned}
x(E_\varphi[U]) &= x(F) && \text{by (30.15)}\\
&\le |V| - \kappa(F) && \text{by (29.7c)}\\
&\le |V| - \big(|V| - |U| + 1\big) && \text{by (30.16)}\\
&= |U| - 1.
\end{aligned}$$

Thus, (30.13b) holds, and thus x is feasible for (30.13).

Let x be feasible for (30.13). As above, to prove that x is feasible for (29.7), it suffices to show that (29.7c) holds. Let $F \subseteq E$. Let $\mathcal{C}$ be the set of components of $(V, F, \varphi\restriction_F)$, so that

$$\kappa(F) = |\mathcal{C}|. \tag{30.17}$$

Clearly, each $C \in \mathcal{C}$ is a subgraph of $G[C]$, so

$$E(C) \subseteq E_\varphi[C] \qquad \text{for each } C \in \mathcal{C}. \tag{30.18}$$

Since the union of the pairwise disjoint sets $\{\, E(C) : C \in \mathcal{C}\}$ is F, we find that

$$\begin{aligned}
x(F) &= \sum_{C\in\mathcal{C}} x\big(E(C)\big)\\
&\le \sum_{C\in\mathcal{C}} x\big(E_\varphi[C]\big) && \text{by (30.18) since } x \ge 0\\
&\le \sum_{C\in\mathcal{C}} \big(|C| - 1\big) && \text{by (30.13c)}\\
&= \Big(\sum_{C\in\mathcal{C}} |C|\Big) - |\mathcal{C}|\\
&= |V| - \kappa(F) && \text{by (30.17)},
\end{aligned}$$

i.e., (29.7c) holds. □

[i] This is a standard graph-theoretic definition, which we shall skip.

Appendix A. Notation Tables

Table 1. Notation for special sets.

B^A	$:=$	the collection of all functions from a set A to a set B
$\mathbb{N}$	$:=$	the set of natural numbers; note that $0 \in \mathbb{N}$
$[n]$	$:=$	$\{1, \ldots, n\}$ for each $n \in \mathbb{N}$
$\mathbb{Z}$	$:=$	the set of integers
$\mathbb{Q}$	$:=$	the set of rationals numbers
$\mathbb{R}$	$:=$	the set of real numbers
$\mathbb{R}^n$	$:=$	$\mathbb{R}^{[n]}$; see §10.1
$\mathbb{R}_+$	$:=$	$\{ x \in \mathbb{R} : x \geq 0\}$, the set of nonnegative reals
$\mathbb{R}_{++}$	$:=$	$\{ x \in \mathbb{R} : x > 0\}$, the set of positive reals
$\mathbb{R}_-$	$:=$	$\{ x \in \mathbb{R} : x \leq 0\}$, the set of nonpositive reals
$\mathbb{Q}_+$	$:=$	$\{ x \in \mathbb{Q} : x \geq 0\}$, the set of nonnegative rationals
$\mathbb{Z}_+$	$:=$	$\{ x \in \mathbb{Z} : x \geq 0\}$, the set of nonnegative integers
$\binom{V}{k}$	$:=$	the collection of k-subsets of a set V

Table 2. Notation for an optimization problem $\mathcal{O} = (X, f, \alpha)$.

$\mathrm{OPT}(\mathcal{O})$	$:=$	the optimal value of $\mathcal{O}$
$L_{\mathcal{O}}(\mu)$	$:=$	the μ-level set of $\mathcal{O}$ for $\mu \in \mathbb{R}$; see §6.1
$\mathcal{O} \overset{\varphi}{\to} \mathcal{P}$	$:=$	denotes that φ is a homomorphism from $\mathcal{O}$ to the optimization problem $\mathcal{P}$
$\mathcal{O} \to \mathcal{P}$	$:=$	denotes that there is a homomorphism from $\mathcal{O}$ to the optimization problem $\mathcal{P}$

TABLE 3. Notation for a digraph $D = (V, A, \varphi)$.

$V(D)$	$\coloneqq$	the vertex set of D; also, V_D
$A(D)$	$\coloneqq$	the arc set of D; also, A_D
$V(W)$	$\coloneqq$	$\{v_0, \dots, v_\ell\}$ for a walk $W = \langle v_0, a_1, v_1, \dots, a_\ell, v_\ell\rangle$
$A(W)$	$\coloneqq$	$\{a_1, \dots, a_\ell\}$ for a walk $W = \langle v_0, a_1, v_1, \dots, a_\ell, v_\ell\rangle$
WZ	$\coloneqq$	the concatenation of the walks W and Z, as in (2.2)
$c(W)$	$\coloneqq$	$\sum_{i=1}^{\ell} c(a_i)$, the cost of a walk $W = \langle v_0, a_1, v_1, \dots, a_\ell, v_\ell\rangle$, where $c\colon A \to \mathbb{R}$ is a cost function
$\mathbb{1}_W$	$\coloneqq$	the vector of the walk W; see §11.9
$\rightsquigarrow_D$	$\coloneqq$	the reachability (binary) relation on V; see §2.2
uv	$\coloneqq$	(u, v), abbreviation for an ordered pair
$(u, v)^{-1}$	$\coloneqq$	(v, u) for an ordered pair (u, v)
D^{-1}	$\coloneqq$	the reverse of D, as in §3.9
$\delta^{\text{out}}_D(U)$	$\coloneqq$	$\{\, a \in A : \varphi(a) \in R \times \overline{R}\}$ for $R \subseteq V$
$\delta^{\text{in}}_D(U)$	$\coloneqq$	$\{\, a \in A : \varphi(a) \in \overline{R} \times R\}$ for $R \subseteq V$
$\delta^{\text{out}}_D(v)$	$\coloneqq$	$\delta^{\text{out}}_D(\{v\})$ for $v \in V$
$\delta^{\text{in}}_D(v)$	$\coloneqq$	$\delta^{\text{in}}_D(\{v\})$ for $v \in V$
$A_\varphi[U]$	$\coloneqq$	the set of arcs induced by $U \subseteq V$; see §4.7
$D[U]$	$\coloneqq$	the subdigraph of D induced by $U \subseteq V$; see §4.7
B_D	$\coloneqq$	the $(V \times A)$ incidence matrix of D; see §11.8
dist_D	$\coloneqq$	the distance function of D; see §15.10

TABLE 4. Notation for a digraph $D = (V, A, \varphi)$, a capacity function $u\colon A \to \mathbb{R}_+ \cup \{+\infty\}$, and a vector $f \in \mathbb{R}^A_+$.

excess_f	$\coloneqq$	$B_D f$, the excess of $f \in \mathbb{R}^A$; see (12.5)
$\text{value}(f)$	$\coloneqq$	the value of an (r, s)-flow f; see §11.13
$D(f, u)$	$\coloneqq$	the residual digraph of f in D with respect to u; see §14.3
d_P	$\coloneqq$	the direction vector from an augmenting path P; see §14.3
u_f	$\coloneqq$	the residual capacities for u relative to f; see (14.10)

TABLE 5. Notation for a graph $G = (V, E, \varphi)$.

$V(G)$	$\coloneqq$	the vertex set of G; also, V_G
$E(G)$	$\coloneqq$	the edge set of G; also, E_G
$V(W)$	$\coloneqq$	$\{v_0, \dots, v_\ell\}$ for a walk $W = \langle v_0, e_1, v_1, \dots, e_\ell, v_\ell\rangle$
$E(W)$	$\coloneqq$	$\{e_1, \dots, e_\ell\}$ for a walk $W = \langle v_0, e_1, v_1, \dots, e_\ell, v_\ell\rangle$
WZ	$\coloneqq$	the concatenation of the walks W and Z, as in (6.8)
uv	$\coloneqq$	$\{u, v\}$, abbreviation for an unordered pair
$\sim_G$	$\coloneqq$	the reachability (binary) relation V; see §6.11
$E_\varphi[U]$	$\coloneqq$	the set of edges induced by $U \subseteq V$; see §6.9
$G[U]$	$\coloneqq$	the subgraph of D induced by $U \subseteq V$; see §6.9
$G - S$	$\coloneqq$	the graph obtained by deletion of S, with $S \subseteq V$ or $S \subseteq E$; see §25.1
G/S	$\coloneqq$	the graph obtained by the identification of S, with $S \subseteq V$; see §25.6
$\delta_G(S)$	$\coloneqq$	the cut induced by S; see §9.4
$\delta_G(v)$	$\coloneqq$	$\delta_G(\{v\})$; see §9.4
V_M	$\coloneqq$	the set of vertices that are saturated by a matching M; see §7.1
$\nu(G)$	$\coloneqq$	the matching number of G; see Problem 7.2
$\tau(G)$	$\coloneqq$	the vertex cover number of G; see §8.1
A_G	$\coloneqq$	the adjacency matrix of G; see (10.13)
B_G	$\coloneqq$	the $(V \times E)$ incidence matrix of G; see (10.14)
$\mathrm{Odd}(G)$	$\coloneqq$	the set of odd components of G
$\mathrm{odd}(G)$	$\coloneqq$	the number of odd components of G

TABLE 6. Notation for vectors and matrices.

x_v	$\coloneqq$	$x(v)$, the vth entry of a vector $x \in \mathbb{R}^V$, with $v \in V$
$\mathbb{1}$	$\coloneqq$	the all-ones vector in the appropriate space
e_v	$\coloneqq$	the vth canonical basis vector of $\mathbb{R}^V$; see (10.4)
$\mathbb{1}_U$	$\coloneqq$	the incidence vector of $U \subseteq V$ in $\mathbb{R}^V$; see (10.9)
$x(U)$	$\coloneqq$	$\mathbb{1}_U^\mathsf{T} x$, for $x \in \mathbb{R}^V$ and $U \subseteq V$; see (10.12)
A_{uv}	$\coloneqq$	$A(uv)$, the uvth entry of a matrix $A \in \mathbb{R}^{U \times V}$, with $uv \in U \times V$
0	$\coloneqq$	the zero matrix in the appropriate space
A^T	$\coloneqq$	the transpose of a matrix $A \in \mathbb{R}^{U \times V}$; see (10.18)
$\geq$	$\coloneqq$	the vector ordering defined in §10.12
$\mathrm{supp}(x)$	$\coloneqq$	$\{ v \in V : x_v \neq 0\}$, the support of a vector $x \in \mathbb{R}^V$; see §18.4

TABLE 7. Other notation.

$[P]$	$:=$	1 if the predicate P holds and *strongly zero* otherwise; this is Iverson's notation
$f(A')$	$:=$	$\{ f(a) : a \in A' \}$, the image of A' under $f\colon A \to B$, where $A' \subseteq A$
$f^{-1}(B')$	$:=$	$\{ a \in A : f(a) \in B' \}$, the preimage of B' under $f\colon A \to B$, where $B' \subseteq B$
$\sup A$	$:=$	the supremum of a subset $A \in \mathbb{R}$, i.e., the least upper bound in $[-\infty, +\infty]$ for the numbers in A
$\inf A$	$:=$	the infimum of a subset $A \in \mathbb{R}$, i.e., the greatest lower bound in $[-\infty, +\infty]$ for the numbers in A
$f\restriction_S$	$:=$	$f \cap (S \times B)$, the restriction of a function $f\colon A \to B$ to a subset $S \subseteq A$
$\perp$	$:=$	"bottom", more popularly known as NIL
$\bigsqcup_{i \in I} A_i$ p	$:=$	$\bigcup_{i \in I} \{i\} \times A_i$, the *disjoint union* of the family $(A_i)_{i \in I}$
$A_1 \sqcup \cdots \sqcup A_k$	$:=$	$\bigcup_{i \in [k]} \{i\} \times A_i$, the *disjoint union* of the family $i \in [k] \mapsto A_i$
$V/\!\sim$	$:=$	the collection of equivalence classes of equivalence relation $\sim$ on V; see §6.10
$A \,\Delta\, B$	$:=$	$(A \setminus B) \cup (B \setminus A)$, the *symmetric difference* of the sets A and B
$(v, w)^\alpha$	$:=$	either (v, w) or (w, v) in the obvious way, for every ordered pair (v, w) and $\alpha \in \{\pm 1\}$; see §14.3

Acknowledgments

The author is thankful for typo corrections and suggestions for improvements from:

- Jonas Rodrigues Lima Gonçalves
- Pedro Gigeck Freire
- Thiago Lima Oliveira

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows*. Theory, algorithms, and applications. Prentice Hall, Inc., Englewood Cliffs, NJ, 1993, pages xvi+846 (cited on page 52).

[2] W. J. Cook, W. H. Cunningham, W. R. Pulleyblank, and A. Schrijver. *Combinatorial optimization*. Wiley-Interscience Series in Discrete Mathematics and Optimization. A Wiley-Interscience Publication. John Wiley & Sons, Inc., New York, 1998, pages x+355 (cited on pages 52, 133).

[3] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*. 3rd edition. MIT Press, Cambridge, MA, 2009, pages xx+1292 (cited on pages 11, 33).

[4] J. Edmonds. "Paths, trees, and flowers". In: *Canad. J. Math.* 17 (1965), pages 449–467 (cited on pages 29, 122, 123).

[5] P. R. Halmos. *Naive set theory*. Reprint of the 1960 edition, Undergraduate Texts in Mathematics. Springer-Verlag, New York-Heidelberg, 1974, pages vii+104 (cited on page 4).

[6] M. Jünger, G. Reinelt, and G. Rinaldi, editors. *Combinatorial optimization—Eureka, you shrink!* Volume 2570. Lecture Notes in Computer Science. Papers dedicated to Jack Edmonds, Papers from the 5th International Workshop held in Aussois, March 5–9, 2001. Springer-Verlag, Berlin, 2003, pages x+209 (cited on page 122).

[7] A. Nemirovski. *Optimization II: Numerical Methods for Nonlinear Continuous Optimization*. 1999. URL: http://www2.isye.gatech.edu/~nemirovs/Lect_OptII.pdf (visited on 10/16/2020) (cited on page 79).

[8] G. Pap. "A constructive approach to matching and its generalizations". PhD thesis. Eötvös Loránd University, 2006 (cited on page 123).

[9] A. Schrijver. *Combinatorial optimization. Polyhedra and efficiency*. Volume 24. 3 volumes. Berlin: Springer-Verlag, 2003. xxxviii+1881 (cited on pages 72, 94, 139).